(12)特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2005 年9 月22 日 (22.09.2005)

PCT

(10) 国際公開番号
WO 2005/088479 A1

(51) 国際特許分類[7]: G06F 17/30

(21) 国際出願番号: PCT/JP2005/004190

(22) 国際出願日: 2005 年3 月10 日 (10.03.2005)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2004-075404 2004 年3 月16 日 (16.03.2004) JP

(71) 出願人 (米国を除く全ての指定国について): 株式会社 ターボデータラボラトリー (TURBO DATA LABORATORIES INC.) [JP/JP]; 〒2210005 神奈川県横浜市神奈川区松見町四丁目１１０１番地７ Kanagawa (JP).

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 古庄 晋二 (FURUSHO, Shinji) [JP/JP]; 〒2210005 神奈川県横浜市神奈川区松見町４丁目１１０１番地７コートハウス菊名８０４号 Kanagawa (JP).

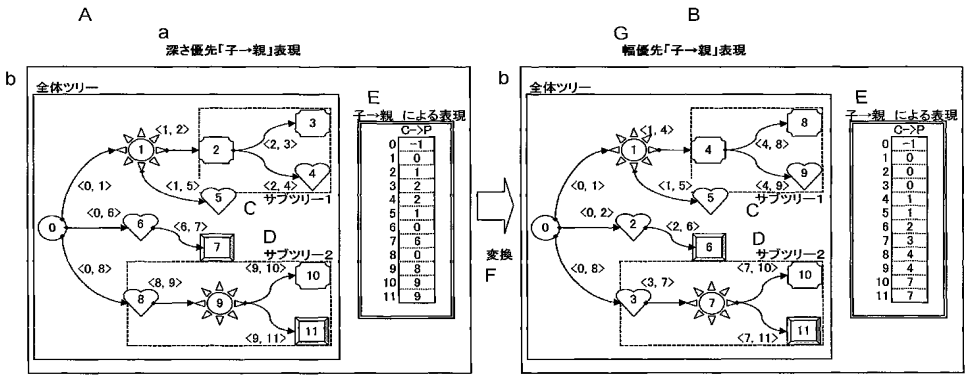
(74) 代理人: 吉田 聡 (YOSHIDA, Satoshi); 〒2330001 神奈川県横浜市港南区上大岡東２−２４−４ Kanagawa (JP).

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NA,

[続葉有]

(54) Title: METHOD FOR HANDLING TREE-TYPE DATA STRUCTURE, INFORMATION PROCESSING DEVICE, AND PROGRAM

(54) 発明の名称: ツリー型データ構造を取り扱う方法、情報処理装置、及び、プログラム



a..."CHILD→PARENT"EXPRESSION WITH DEPTH PRIORITY
b ... WHOLE TREE
C... SUB TREE 1
D... SUB TREE 2
E... EXPRESSION BY CHILD→PARENT
F... CONVERSION
G..."CHILD→PARENT"EXPRESSION WITH WIDTH PRIORITY

(57) Abstract: It is possible to express a tree-type data structure so as to effectively trace the relationship between data in the tree-type data structure (for example, parent-child, ancestor, descendant, brothers, generations). In the tree-type data structure, for each of non-route nodes which are nodes excluding the route nodes, their parent nodes are correlated so that the parent-child relationship between the nodes is expressed by using the "child → parent" relationship. Accordingly, by specifying a child node, it is possible to promptly specify the only one parent node corresponding to the child node.

(57) 要約: ツリー型データ構造のデータ間の関係（例えば、親子、祖先、子孫、兄弟、世代）を効率的にトレースすることができるようにツリー型データ構造を表現する。ツリー型データ構造において、ルート・ノード以外のノードである非ルート・ノードの各々に対して該非ルート・ノードの親ノードを関連付けることにより、ノード間の親子関係

[続葉有]

WO 2005/088479 A1

NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:
— 国際調査報告書

２文字コード及び他の略語については、定期発行される各*PCT*ガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

が「子→親」関係を用いて表現されるので、子ノードを特定することによって、この子ノードに対応する唯一の親ノードが直ちに特定される。

明 細 書

ツリー型データ構造を取り扱う方法、情報処理装置、及び、プログラム

技術分野

[0001] 本発明はツリー型データ構造を取り扱う方法、特に、ツリー型データ構造を表現し、記憶装置上に構築し、又は、変換する方法に関する。また、本発明は、かかる方法を実施する情報処理装置に関する。更に、本発明は、かかる方法を実行するためのプログラム、及び、このプログラムを記録した記録媒体に関する。

背景技術

[0002] データベースは種々の用途に用いられているが、中規模ないし大規模システムにおいては、論理的な矛盾が排除できるリレーショナルデータベース(RDB)の使用が主流となっている。たとえば、RDBは飛行機の座席予約等のシステムに利用されている。この場合、キー項目を指定することにより、(多くの場合1件の)ターゲットを迅速に検索することもでき、或いは、予約の確定、キャンセル或いは変更などを行うことができる。また、各便の座席数はせいぜい数百であるため、特定の航空便の空席数を求めることも可能である。

[0003] このようなRDBは、表形式データの取り扱いに適しているが、ツリー形式データの取り扱いには適していないことが知られている(例えば、非特許文献1を参照。)。

[0004] 更に、アプリケーションの中には、表形式による表現よりもツリー形式による表現の方が適しているものが存在する。特に、近年、イントラネットやインターネットのアプリケーションのデータ標準として、ツリー型データ構造を採用するXMLが普及している(XMLの詳細については、例えば、非特許文献2を参照。)。

[0005] しかし、ツリー型データ構造の取り扱い、例えば、ツリー形式データの検索は、一般に、大変効率が悪い。この効率の悪さの第1の理由は、データが各所のノードに分散して存在するため、データの存在すべき場所を直ちに特定することが困難である点にある。RDBでは、例えば、「年齢」というデータは、あるテーブルの「年齢」という項目だけに格納されている。しかし、ツリー型データ構造では、「年齢」というデータを保持するノードが各所に散在しているので、一般的には、ツリー型データ構造の全体を

調べなければ、該当するデータを検索することができない。

[0006] 効率の悪さの第2の理由は、検索の結果を表現するために時間がかかるという点にある。検索にヒットしたノード群を表現しようとすると、屡々、そのノードの子孫にあたるノードも表現しなければならないが、RDBMSとは異なりデータ構造が非定型であるため、子孫ノードを表現するために時間がかかる。

[0007] そこで、データベースの主流であるRDBの利点をいかすため、従来、ツリー型データ構造をデータベース化するとき、ツリー形式データをRDB化する方法(例えば、特許文献1を参照。)が提案されている。RDBでは、データはテーブル(表)に分解して保持される。そのため、実際のツリー形式データをRDB化するには、ツリー形式データをテーブルに押し込める必要がある。しかし、様々のツリー型データ構造を取り扱うためには、その構造毎に個別にデータをテーブルに押し込め、システム設計を行わなければならない。したがって、RDBに基づくシステム構築は非常に手間のかかる作業である。

[0008] これに対して、ツリー形式データ、特に、XMLデータをそのままの形でデータベース化する方法も提案されている。ツリー型データ構造の場合、一つのノードに子孫ノードをぶら下げることができ、多様な表現が可能であるため、システム設計の手間を大幅に削減することができる。したがって、XMLのようなツリー構造を取り扱える技術を核として、ツリー構造データを処理することへのニーズが高まっている。

[0009] XMLデータをそのままの形でデータベース化する方法の一例のアプローチは、ツリー構造に記入されているデータのコピーを取り出し、例えば、「年齢」という項目であれば、「年齢」の検索用インデックスデータを別途保持する(例えば、特許文献2を参照。)。これにより、データ自身に属性を付加できるというXMLデータのメリットを十分に活用すると共に、タグを用いて表現された各項目の関係構造をそのまま記憶できるようにしている。

特許文献1:特開2003−248615号公報

特許文献2:特開2001−195406号公報

非特許文献1:株式会社セック、"Karearea WhitePaper"、[online]、[平成16年2月19日検索]、インターネット<URL:http://www.sec.co.jp/products/karearea/>

非特許文献2:W3C、"Extensible Markup Language (XML) 1.0 (Third Edition)"、[online]、2004年2月4日、[平成16年2月19日検索]、インターネット<URL:http://www.w3.org/TR/2004/REC-xml-20040204/>

発明の開示

発明が解決しようとする課題

[0010] しかし、検索用インデックスデータを別途保持するようなアプローチでは、少なくともデータは二重に保持され、かつ、インデックスを作成するコスト及びインデックスを格納するためのデータ領域が必要となり、大規模なデータを保持する上で不利である。

[0011] 実際、このようなメカニズムによって、実際に検索を行い、ノードを特定したとしても、そのノードを表現するためには時間がかかる。また、このメカニズムは、ノード間の関係を問題とする検索(例えば、祖先に「60歳」という「年齢」を含み、子孫に「1歳」という「年齢」を含むツリーの抽出)には利用できない。

[0012] このような従来技術の根本的な問題点は、個々のデータのみに着目し、データを蓄えたノード間をポインタで接続することによりツリー型データ構造が表現されているため、データ間の関係、例えば、親子、祖先、子孫、兄弟(シブリング)、世代などの関係を効率的にトレースすることができないことにある。換言すると、ポインタは、その値が一定しないため、データの格納アドレスを示すという用途にしか使用できず、ノード間の関係を直接的に表現することができない。

[0013] そこで、本発明は、ツリー型データ構造のデータ間の関係を効率的にトレースすることができるツリー型データ構造の表現、構築、及び、変換方法の提供を目的とする。

[0014] 更に、本発明は、ツリー型データ構造のデータ間の関係を効率的にトレースすることができるツリー型データ構造を構築、及び、変換する情報処理装置の提供を目的とする。

[0015] 更に、本発明は、ツリー型データ構造のデータ間の関係を効率的にトレースすることができるツリー型データ構造の表現、構築、及び、変換プログラムの提供を目的とする。

[0016] 更に、本発明は、上記ツリー型データ構造の表現、構築、及び、変換プログラムを

記録した記録媒体の提供を目的とする。

課題を解決するための手段

[0017] 上記目的を達成するため、本発明の原理は、ツリー型データ構造を構成するノード間の親子関係を、親ノードに子ノードを対応付ける「親→子」関係ではなく、子ノードに親ノードを対応付ける「子→親」関係によって表現することである。

[0018] 「親→子」関係によって親子関係を表現する場合、一つの親ノードに複数の子ノードが対応する場合があるので、親ノードと子ノードの二つの要素を特定しなければ親子関係を定義できない。即ち、親ノードを特定しても、その親ノードと親子関係にある子ノードを特定することができない。

[0019] これに対して「子→親」関係によって親子関係を表現する場合、一つの子ノードには必ず唯一の親ノードが対応するので、子ノードを特定することによって、この子ノードに対応する唯一の親ノードを直ちに特定することができる。

[0020] そのため、本発明によれば、ツリー型データ構造を構成するノード間の親子関係を記憶装置上で表現する方法は、ルート・ノード以外のノードである非ルート・ノードの各々に対して、該非ルート・ノードの親ノードを関連付けることにより前記ノード間の親子関係を表現する。これにより、「子→親」表現された子のノードから親のノードのリストを辿ることでツリーのトポロジーを表現することができる。

[0021] また、本発明によれば、ツリー型データ構造を記憶装置上に構築する方法は、

ルート・ノードを含むノードに固有のノード識別子を付与するノード定義ステップと、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与されたノード識別子に、前記非ルート・ノードの各々の親ノードに付与されたノード識別子を関連付ける親子関係定義ステップと、

を含む。このように、最初に、文字列、浮動小数、整数などの任意の識別情報によってノードにノード識別子を付与し、次に、「子→親」表現に基づいて親子関係を定義することによって、子ノードのノード識別子から親ノードのノード識別子を引く(ルックアップする)ことでツリーのトポロジーを表現することができる。

[0022] 上記の方法において、前記ノード定義ステップは前記ノード識別子として数値を用いる。ノード識別子を特に数値で表すことによって、親のノード識別子の格納アドレス

を特定することが容易になるので、より簡単に子のノード識別子から親のノード識別子を引くことができるようになる。

[0023] また、上記の方法において、前記ノード定義ステップは前記ノード識別子として連続する整数を用いる。ノード識別子を連続する整数で表すことによって、ノード識別子から、そのノードに対応する親ノードのノード識別子が格納されているアドレスを簡単に取得することができるので、子ノードのノード識別子から親ノードのノード識別子を引く処理を高速化することができる。

[0024] 好ましい実施例によれば、ノード識別子は、0または1からの整数連番で表現される。

[0025] 上記の本発明によるツリー型データ構造を記憶装置上に構築する方法において、前記ノード間の親子関係は、前記非ルート・ノードの各々に関連付けられた前記親ノードの配列によって表現される。この結果として、配列を利用することにより、子ノードから親ノードを引く処理が高速化される。

[0026] また、前記親子関係定義ステップは、前記非ルート・ノードの各々に付与されたノード識別子に関連付けられた前記親ノードに付与されたノード識別子の配列を前記記憶装置に格納するようにしてもよい。この結果として、親ノードのノード識別子が格納されているアドレスを簡単に取得することができるようになるので、子ノードから親ノードを引く処理が高速化される。

[0027] ツリー型データ構造のノードにノード識別子として順序付きの番号を付与してノード間の親子関係を表現する場合、番号の付与順序に規則を定めることによって、その後のツリー型データ構造の取り扱いが容易になるという利点がある。本発明によれば、この番号の付与順序の規則として、同じ世代のノードよりも子ノードを優先する深さ優先モードと、子ノードよりも同じ世代のノードを優先する幅優先モードが利用される。

[0028] このため、本発明によれば、ツリー型データ構造を記憶装置上に構築する方法は、

同じ世代のノードよりも子ノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義ステップと、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の

順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義ステップと、

を含む。これにより、ノードは深さ優先で連続整数が付与され、ノード間の親子関係は「子→親」関係の配列によって表現される。深さ優先で連続番号が付与されたノードの親子関係を「子→親」関係に基づいて配列表現すると、あるノードの子孫ノードが連続領域に出現するという優れた性質が得られる。

[0029] 本発明の好ましい一実施例では、上記の方法において、

前記ノード定義ステップは、

最初にルート・ノードに番号を付与するステップと、

既に番号が付与されたあるノードに唯一の子ノードが存在する場合には、当該子ノードに当該あるノードに付与された前記番号の次の番号を付与するステップと、

既に番号が付与されたあるノードに複数の子ノードが存在する場合には、当該複数の子ノードの間の兄弟関係に従って、弟ノードは直上の兄ノードの全ての子孫ノードに番号が付与された後に次の番号が付与されるように、一番上の兄ノードから一番下の弟ノードまで番号を付与するステップと、

を含む。これにより、深さ優先モードで同一の親ノードから派生した複数の子ノードの間に兄弟関係が定義される。

[0030] 更に、本発明によれば、ツリー型データ構造を記憶装置上に構築する方法は、

子ノードよりも同じ世代のノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義ステップと、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義ステップと、

を含む。これにより、ノードは幅優先モードで連続整数が付与され、ノード間の親子関係は「子→親」関係の配列によって表現される。幅優先モードで連続番号が付与されたノードの親子関係を「子→親」関係に基づいて配列表現すると、親ノードに付与された番号が前記配列中に順序付き(昇順又は降順)で出現するという優れた性質が得られる。

[0031] 本発明の好ましい一実施例では、上記の方法において、

前記ノード定義ステップは、

各ノードが前記ルート・ノードから何世代目のノードであるか、及び、各世代に含まれるノード数を算出するステップと、

最初に前記ルート・ノードに番号を付与するステップと、

ある世代に含まれる全てのノードに番号が付与されたならば、当該ある世代の次の世代にノードが存在しなくなるまで、当該次の世代に含まれる全てのノードに対して、親ノードが異なる場合には、当該親ノードに番号が付与された順番に当該ノードに番号を付与し、当該親ノードが同一である場合には、当該親ノードから派生した複数の子ノードの間に兄弟関係を定義し、一番上の兄ノードから一番下の弟ノードまで直前に付与された番号の次の番号から連続的に変化する固有の整数を順に付与するステップと、

を含む。これにより、幅優先モードで同一の親ノードから派生した複数の子ノードの間に兄弟関係が定義される。

[0032] 本発明の一実施例では、前記深さ優先モードの優れた性質を利用することにより、上記の方法は、前記配列から、あるノードに付与された整数以上の値が格納されている連続領域を抽出することにより、前記あるノードの全ての子孫ノードを特定するステップを更に有する。これにより、あるノードの子孫ノードを表すノード群が前記配列内の連続ブロックとして獲得できる。

[0033] また、本発明の他の一実施例では、前記幅優先モードの優れた性質を利用することにより、上記の方法は、前記配列から、あるノードに付与された整数と同じ値が格納されている連続領域を抽出することにより、前記あるノードの全ての子ノードを特定するステップを更に有する。これにより、あるノードの子ノードを、例えば、二分探索で検索することが可能であり、即ち、O(log(n))のオーダーで検索することが可能になる。

[0034] 上述のように、ノードに連続番号を付与するための深さ優先モード及び幅優先モードは、それぞれ、固有の優れた性質を備えている。そこで、本発明によれば、ツリー型データ構造を記憶装置上に構築する方法は、

ルート・ノードから始めて全てのノードに連続的に変化する整数を一意に割り当てる

ステップと、

ノード間に親子関係を定義するステップと、

を含み、

前記全てのノードに整数を一意に割り当てるステップは、

同じ世代のノードよりも子ノードを優先して番号を付与する深さ優先モードと、子ノードよりも同じ世代のノードを優先して番号を付与する幅優先モードのどちらのモードでノードに番号を付与するかを選択するステップと、

前記深さ優先モードが選択された場合に、深さ優先でノードを検索し、検索された順にノードに番号を付与するステップと、

前記幅優先モードが選択された場合に、幅優先でノードを検索し、検索された順にノードに番号を付与するステップと、

を含み、

前記ノード間に親子関係を定義するステップは、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納するステップを含む。これにより、深さ優先モードによるノード番号付与と幅優先モードによるノード番号付与を一つのシステムに併存させることができるので、状況に応じて適切な表現形式を利用することが可能である。

[0035] また、本発明の一実施例によれば、上記の方法において、

前記ノード間に親子関係を定義するステップは、

子ノードから親ノードへの関係を定義する子親表現モードと、親ノードから子ノードへの関係を定義する親子表現モードのどちらのモードで親子関係を定義するかを選択するステップと、

前記子親表現モードが選択された場合に、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納するステップと、

前記親子表現モードが選択された場合に、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号の配列を前記記憶装置に格納するステップと、

を含む。これにより、「子→親」関係で表現されているノード間の親子関係を、状況に応じて、「親→子」関係で表現することが可能になる。「親→子」関係に基づく表現は、例えば、外部との情報交換の際に有利である。

[0036] 上述のように、ツリー型データ構造の表現形式として、親子関係を表現するための「子→親」表現及び「親→子」表現、並びに、ノードに番号を付与するため深さ優先モード及び幅優先モードを選択的に利用可能である。そこで、本発明は、異なる表現形式の相互変換の方法を提供する。

[0037] 本発明によれば、記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法は、

前記親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより定義され、

同じ世代のノードよりも子ノードを優先してノードに番号を付与する深さ優先モードで表現されたツリー型データ構造の各ノードの世代を判定し、世代毎に属しているノードの個数を計数するステップと、

前記世代毎に属しているノードの個数に基づいて、子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで番号を付与する際に各世代において付与される番号を決定するステップと、

前記判定されたノードの世代及び前記決定された各世代において付与される番号に基づいて、前記各ノードの番号を前記幅優先モードで付与される番号に変換する変換配列を作成するステップと、

前記各ノードの前記親子関係を、前記変換配列を使用して前記幅優先モードで付与される番号で表現された親子関係に変換するステップと、

を有する。これにより、深さ優先モードに基づく「子→親」表現形式から幅優先モードに基づく「子→親」表現形式への変換が可能になる。

[0038] また、本発明によれば、記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法は、

前記親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより定義され、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードの子孫の数を計数するステップと、

親ノードに付与されるべき番号に、当該ノードと同一の親ノードから派生した兄弟ノードのうち当該ノードよりも前に番号が付与されている兄ノードの数及び当該兄ノードの子孫の数を加算することにより、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成するステップと、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換するステップと、

を有する。これにより、幅優先モードに基づく「子→親」表現形式から深さ優先モードに基づく「子→親」表現形式への高速変換が可能になる。

[0039] また、本発明によれば、記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法は、

前記親子関係が、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより定義され、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードを深さ優先で検索し、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成するステップと、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換するステップと、

を有する。これにより、幅優先モードに基づく「子→親」表現形式から深さ優先モードに基づく「子→親」表現形式への検索に基づく変換が可能になる。「深さ優先」検索は、例えば、スタックを使用して番号変換配列を作成することにより実現される。

[0040] また、本発明によれば、記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法は、

前記親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を第1の配列の要素として前記記憶装置に格納することにより

定義され、

各ノードに関して、当該ノードに付与された番号が前記第1の配列の要素として出現する回数を計数するステップと、

前記各ノードに関して当該ノードに対応する子ノードに付与された番号を格納するため、前記計数された回数に応じた個数の連続領域を前記記憶領域に第2の配列として確保するステップと、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する子ノードの番号を順次に格納するステップと、

を有する。これにより、親子関係は、「子→親」表現形式から「親→子」表現形式に変換される。即ち、変換後の親子関係は、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号を前記第2の配列の要素として前記記憶装置に格納することにより定義される。

[0041] 更に、本発明によれば、記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法は、

前記親子関係が、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号を第1の配列の要素として前記記憶装置に格納することにより定義され、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を格納するため、前記記憶装置に第2の配列を確保するステップと、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する親ノードの番号を順次に格納するステップと、

を有する。これにより、親子関係は、「親→子」表現形式から「子→親」表現形式に変換される。即ち、変換後の親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記第2の配列の要素として前記記憶装置に格納することにより定義される。

[0042] また、本発明によれば、上記の本発明の方法を実施する情報処理装置が提供され

る。

[0043] また、本発明によれば、上記の本発明の方法を実行するためのプログラムが提供される。

[0044] 更に、本発明によれば、上記の本発明のプログラムを記録した記録媒体が提供される。

発明の効果

[0045] 本発明によれば、ツリー型データ構造のノード間の親子関係は、「子→親」表現に基づいて記述されるので、一つのノードに対して一つの格納場所を設けることにより親子関係を定義することができる。したがって、ツリー型データ構造を操作する際にアクセスされるメモリの量が低減し、これにより、操作も高速化される。

[0046] 更に、本発明の「子→親」表現によれば、幅優先モードでノードに番号を付与することにより、あるノードから派生した子ノードを容易に検索することができる。

[0047] また、本発明の「子→親」表現によれば、深さ優先モードでノードに番号を付与することにより、あるノードの子孫ノードのブロックを容易に特定することができる。

発明を実施するための最良の形態

[0048] 以下、添付図面を参照して、本発明の実施の形態につき説明を加える。

[0049] [コンピュータシステム構成]

図1は、本発明の実施の形態にかかるツリー型データ構造を取り扱うコンピュータシステムのハードウェア構成を示すブロックダイヤグラムである。図1に示すように、このコンピュータシステム10は、通常のものと同様の構成であり、プログラムを実行することによりシステム全体および個々の構成部分を制御するCPU12、ワークデータなどを記憶するRAM(Random Access Memory)14、プログラム等を記憶するROM(Read Only Memory)16、ハードディスク等の固定記憶媒体18、CD—ROM19をアクセスするためのCD—ROMドライバ20、CD—ROMドライバ20や外部ネットワーク(図示せず)と接続された外部端子との間に設けられたインタフェース(I／F)22、キーボードやマウスからなる入力装置24、CRT表示装置26を備えている。CPU12、RAM14、ROM16、外部記憶媒体18、I／F22、入力装置24および表示装置26は、バス28を介して相互に接続されている。

[0050] 本実施の形態にかかる、ツリー型データ構造を記憶装置上に構築するプログラム、及び、ツリー型データ構造を記憶装置上で変換するプログラムは、CD−ROM19に収容され、CD−ROMドライバ20に読取られても良いし、ROM16に予め記憶されていても良い。また、いったんCD−ROM19から読み出したものを、外部記憶媒体18の所定の領域に記憶しておいても良い。或いは、上記プログラムは、ネットワーク(図示せず)、外部端子およびI／F22を経て外部から供給されるものであっても良い。

[0051] また、本発明の実施の形態にかかる情報処理装置は、コンピュータシステム10にツリー型データ構造を記憶装置上に構築するプログラム、及び、ツリー型データ構造を記憶装置上で変換するプログラムを実行させることにより実現される。

[0052] [ツリー型データ構造]

図2A、Bは、ツリー形式データの一例であるPOSデータの説明図であり、図2Aは、このツリー形式データのデータ構造(即ち、トポロジー)及びデータ値を視覚的に表現した一例であり、図2Bは、同じツリー形式データをXML形式で表現した一例である。同図に示されるようにツリー型データ構造は、ルート・ノード(本例では、POSデータ)から始めて、各ノードで枝分かれしてリーフ・ノード(端点)に至るノードとアークの組み合わせによって表現される。各ノードの実体的な値、例えば、店名ノードの値＝″フランス店″の格納場所は、店名ノードに関連したポインタで指定される。

[0053] 本発明は、ツリー型データ構造のトポロジーを対象とするため、以下の説明では、主として、ツリー型データ構造のトポロジーに関して説明する。

[0054] 従来、このようなツリー型データ構造は、データを蓄えたノード間をポインタで接続することによって表現されている。しかし、ポインタ表現は、ポインタ値に必然性がないという欠点がある。即ち、ある場合には特定のノードAがある番地(例えば、100番地)に格納され、別の場合には同じノードAが別の番地(例えば、200番地)に格納されるので、ポインタ値が一定ではなく、ポインタ値は、本質的にノードの格納アドレスを表現するに過ぎない。そのため、例えば、ノードが深さ優先の規則に従ってポインタで接続されている場合、これらのノードを幅優先の規則に従ってポインタで再接続することは困難である。

[0055] これに対して、本発明者は、ツリー型データ構造のトポロジーがアークリストによって

記述可能であることに着目した。アークリストとは、ノード間の親子関係を表すアークのリストである。図3AーCは、アークリストを用いたツリー型データ構造の表現形式の一例の説明図である。図3AーCの例では、0、10、20、30、40、50、60、70、80、90、100及び110のノード識別子(ID)が付与された12個のノードからなるツリー型データ構造が示されている。図3Aはツリー型データ構造の全体を示している。図3Aにおいて、丸形、ハート形などの図形の中央に記載された数字は、ノードIDを表し、矢印と矢印の側に記載された<0，10>などの数字の対は、アークを表している。尚、ノードIDは、文字列には限られず、数値、特に、整数でもよい。図3Bは、親ノード(From−ID)から子ノード(To−ID)へのアークリストを示し、図3Cは、ノードIDとノードTypeの対のリストからなるノードリストを示す。尚、ツリー型データ構造を表現するだけの目的のためにはノードリストが無くても構わない。原理的には、このようなアークリストを用いることによって、ノード間の関係をポインタによらずに直接的に記述することが可能である。

[0056] [「子→親」関係に基づく表現]

図3AーCの例では、アークリストは、親ノードに子ノードを対応付ける「親→子」関係に基づいて記述されている。そのため、一つの親ノード、例えば、ルート・ノード0には、3個の子ノード10、60及び80が存在するため、アークリストのFrom−IDには、同じノードIDの0が3回出現している。つまり、親ノードを特定しても子ノードを特定することができないので、アークリストは、要素From−IDの配列と要素To−IDの配列により構成される。アークリストを使用する場合、あるノードは、From−IDの配列と、To−IDの配列の両方の配列に出現する。

[0057] これに対して、親子関係は、「子→親」関係によっても表現することが可能である。この場合、ノード間の親子関係は、ルート・ノード以外のノードである非ルート・ノードの各々と、関連付けられた親ノードと、の組の配列によって表現される。この「子→親」関係によって親子関係を表現する場合、「親→子」関係の場合には得られなかった重要な性質がある。即ち、一つの子ノードには必ず唯一の親ノードが対応するので、子ノードを特定することによって、この子ノードに対応する唯一の親ノードを直ちに特定することができる。つまり、アークリストは、実際には、要素To−IDの配列だけを準

備すればよい。この結果として、アークリストを格納するための記憶容量が削減される。この記憶容量の削減は、メモリへのアクセス回数が低減するという効果があるので、結果的に、処理の高速化が実現できる。

[0058] 図4A～Cは、本発明の一実施例による「子→親」関係に基づくツリー型データ構造の表現方法の説明図である。図4Aはツリー全体の説明図であり、図4Bは「子→親」関係に基づくアークリストである。図4Bのアークリストは、ルート・ノードに対する親ノードの格納領域を含んでいるので、ルート・ノードの親ノードとして、便宜的に"－"が設定されている。但し、ルート・ノードに対応する親ノードは存在しないので、図4Cに示されるように、「子→親」関係に基づくアークリストからルート・ノードに対する親ノードの格納領域を除いても構わない。このように本発明の一実施例では、ルート・ノード以外のノードである非ルート・ノードの各々に対して、非ルート・ノードの親ノードを関連付けることによりノード間の親子関係を表現する。そして、「子→親」表現された子のノードから親のノードのリストを辿ることでツリーのトポロジーを表現することができる。

[0059] このような「子→親」関係に基づくツリー型データ構造は、本発明の一実施例によれば、図5に示されるように、図1に示されたコンピュータシステム10に、ルート・ノードを含むノードに固有のノード識別子を付与するノード定義ステップ501と、前記ルート・ノード以外のノードである非ルート・ノードの各々に付与されたノード識別子に、前記非ルート・ノードの各々の親ノードに付与されたノード識別子を関連付ける親子関係定義ステップ502と、を実行させることによってRAM14上に構築される。このように、最初に、文字列、浮動小数、整数などの任意の識別情報によってノードにノード識別子を付与し、次に、「子→親」表現に基づいて親子関係を定義することによって、子ノードのノード識別子から親ノードのノード識別子を引く(ルックアップする)ことでツリーのトポロジーを表現することができる。

[0060] ［ノード識別子］

好ましい一実施例によれば、ノード定義ステップはノード識別子として数値を使用し、より好ましくは、連続する整数を使用し、更に好ましくは、0又は1からの整数連番を使用する。これにより、ノード識別子から、そのノードに対応する親ノードのノード識別

子が格納されているアドレスを簡単に取得することができるので、子ノードのノード識別子から親ノードのノード識別子を引く処理を高速化することができる。

[0061] ツリー型データ構造のノードにノード識別子として順序付きの番号を付与してノード間の親子関係を表現する場合、番号の付与順序に規則を定めることによって、その後のツリー型データ構造の取り扱いが容易になるという利点がある。本発明によれば、この番号の付与順序の規則として、同じ世代のノードよりも子ノードを優先する深さ優先モードと、子ノードよりも同じ世代のノードを優先する幅優先モードが利用される。

[0062] 図6A～Cは、本発明の一実施例によりID形式のツリー構造型データを整数連番形式のツリー構造型データへ変換する処理の説明図である。図6Aには、各ノードにID番号が付与されたツリー構造型データが示され、図6Bには、変換規則が示され、図6Cには、各ノードに整数連番が付与されたツリー構造型データが示されている。本例の変換規則は、深さ優先で連続番号を付与する規則であり、具体的には、複数の子ノードが存在する場合、長子（一番上の兄）ノードに最小番号を付与し、末子（一番下の弟）ノードに大きい番号を付与し、かつ、兄弟ノードよりも子ノードを優先して番号を付与する。本例では、昇順に番号付けをしているが、降順に番号付けをしてもよい。

[0063] また、図7A～Cは、本発明の他の一実施例によりID形式のツリー構造型データを整数連番形式のツリー構造型データへ変換する処理の説明図である。図7Aには、各ノードにID番号が付与されたツリー構造型データが示され、図7Bには、変換規則が示され、図7Cには、各ノードに整数連番が付与されたツリー構造型データが示されている。本例の変換規則は、幅優先で連続番号を付与する規則であり、具体的には、複数の子ノードが存在する場合、長子（一番上の兄）ノードに最小番号を付与し、末子（一番下の弟）ノードに大きい番号を付与し、かつ、子ノードよりも兄弟ノードを優先して番号を付与する。本例では、昇順に番号付けをしているが、降順に番号付けをしてもよい。

[0064] このようにノード識別子として番号を使用すると、ノード番号から直ちに、即ち、O（1）のオーダーで、そのノードに関する格納値が格納されているアドレスを引くことがで

きる。また、親子関係を「子→親」表現することによって、子ノードから親ノードを直ちに、即ち、O(1)のオーダーで引くことができる。

[0065] [深さ優先モード]

本発明の一実施例によれば、図6A～Cに示されるような深さ優先に基づくツリー型データ構造は、図1に示されたコンピュータシステム10に、

同じ世代のノードよりも子ノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義ステップと、

ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義ステップと、

を実行させることによって、記憶装置上に構築される。これにより、ノードは深さ優先で連続整数が付与され、ノード間の親子関係は「子→親」関係の配列によって表現される。

[0066] 図8は、本発明の一実施例による深さ優先に基づくノード定義処理のフローチャートである。このノード定義処理は、コンピュータシステム10に

最初にルート・ノードに番号を付与するステップ801と、

既に番号が付与されたあるノードに唯一の子ノードが存在する場合には、当該子ノードに当該あるノードに付与された前記番号の次の番号を付与するステップ802と、

既に番号が付与されたあるノードに複数の子ノードが存在する場合には、当該複数の子ノードの間の兄弟関係に従って、弟ノードは直上の兄ノードの全ての子孫ノードに番号が付与された後に次の番号が付与されるように、一番上の兄ノードから一番下の弟ノードまで番号を付与するステップ803と、

を実行させる。これにより、深さ優先モードで同一の親ノードから派生した複数の子ノードの間に兄弟関係が定義される。

[0067] 図9は、本発明の一実施例により図6A～Cに示された深さ優先のツリー型データ構造から作成された「子→親」表現に基づく親子関係の配列の説明図である。同図にサブツリー1又はサブツリー2として示されているように、深さ優先で連続番号が付与されたノードの親子関係を「子→親」関係に基づいて配列表現すると、あるノードの

子孫ノードが連続領域に出現するという優れた性質が得られる。

[0068] 本発明の一実施例では、深さ優先モードの優れた性質を利用することにより、前記配列から、あるノードに付与された整数以上の値が格納されている連続領域を抽出することにより、前記あるノードの全ての子孫ノードを特定する。これにより、あるノードの子孫ノードを表すノード群が前記配列内の連続ブロックとして獲得できる。例えば、連続ブロックのサイズをmとすると、あるノードの全ての子孫ノードを特定するための処理速度は、O(m)のオーダーになる。

[0069] 既に説明したように、ノード間の親子関係は、「子→親」関係の配列の他に、「親→子」関係の配列によっても表現できる。図10は、図6A～Cに示された深さ優先のツリー型データ構造から作成された「親→子」表現に基づく親子関係の配列の説明図である。一つの親ノードに対して複数の子ノードが存在し得るので、親子関係の配列は、各ノードに対する子ノードの番号が格納されている領域を示すための配列Aggrと、子ノードの番号が格納されている配列P→Cの二つの配列により構成される。例えば、配列Aggrの先頭から2番目の要素Aggr[1]の値は"3"であり、これは、ノード[1]に対する子ノードの番号は、配列P→Cの要素P→C[3]以降に格納されていることを表している。これにより、ノード[0]、即ち、ルート・ノードに対する子ノードは、配列P→Cの先頭から3個の要素、P→C[0]の1、P→C[1]の6、及びP→C[2]の8であることがわかる。

[0070] この「親→子」表現に基づく親子関係の配列の求め方を説明する。

(1)ノードの番号が配列P→Cの最大の添字(＝11)と一致する場合、このノードに属する子ノードは存在しない。したがって、処理は継続されない。

(2)同図に太字で表された親ノードの番号からAggr値を求める。このAggr値は、配列P→Cの開始点を表す。

(3)太字で表された親ノード番号＋1に対応するAggr値を求める。このAggr値－1が配列P→Cの終了点である。

[0071] 例えば、ノード0の子ノードの開始点は、Aggr[0]、即ち、0であり、終了点は、Aggr[1]－1、即ち、3－1＝2である。したがって、ノード0の子ノードは、配列P→Cの0～2番目の要素、即ち、1、6及び8である。

[0072] 或いは、「親→子」表現に基づく親子関係は、より単純に、親ノード番号の配列と、対応する子ノード番号の配列と、の二つの配列により表現することも可能である。しかし、この配列を利用して親子関係を見つけるためには、親ノードの番号を検索しなければならないので、即ち、log(n)のアクセス時間を要するので効率が悪い。

[0073] [幅優先モード]

本発明の一実施例によれば、図7A－Cに示されるような幅優先に基づくツリー型データ構造は、図1に示されたコンピュータシステム10に、

子ノードよりも同じ世代のノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義ステップと、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義ステップと、

を実行させることによって、記憶装置上に構築される。これにより、ノードは幅優先モードで連続整数が付与され、ノード間の親子関係は「子→親」関係の配列によって表現される。

[0074] 図11は、本発明の一実施例による幅優先に基づくノート定義処理のフローチャートである。このノード定義処理は、コンピュータシステム10に、

各ノードが前記ルート・ノードから何世代目のノードであるか、及び、各世代に含まれるノード数を算出するステップ1101と、

最初に前記ルート・ノードに番号を付与するステップ1102と、

ある世代に含まれる全てのノードに番号が付与されたならば、当該ある世代の次の世代にノードが存在しなくなるまで、当該次の世代に含まれる全てのノードに対して、親ノードが異なる場合には、当該親ノードに番号が付与された順番に当該ノードに番号を付与し、当該親ノードが同一である場合には、当該親ノードから派生した複数の子ノードの間に兄弟関係を定義し、一番上の兄ノードから一番下の弟ノードまで直前に付与された番号の次の番号から連続的に変化する固有の整数を順に付与するステップ1013と、

を実行させる。これにより、幅優先モードで同一の親ノードから派生した複数の子ノー

ドの間に兄弟関係が定義される。

[0075] 図12は、本発明の一実施例により図7A～Cに示された幅優先のツリー型データ構造から作成された「子→親」表現に基づく親子関係の配列の説明図である。同図に示されているように、幅優先で連続番号が付与されたノードの親子関係を「子→親」関係に基づいて配列表現すると、あるノードの子ノードは連続領域に出現するという優れた性質が得られる。これは、幅優先モードで連続番号が付与されたノードの親子関係を「子→親」関係に基づいて配列表現すると、親ノードに付与された番号が前記配列中に順序付き(昇順又は降順)で出現することによる。

[0076] したがって、本発明の一実施例では、幅優先モードの優れた性質を利用することにより、前記配列から、あるノードに付与された整数と同じ値が格納されている連続領域を抽出することにより、前記あるノードの全ての子ノードを特定する。これにより、あるノードの子ノードを、例えば、二分探索などの手法を用いて検索することが可能であり、即ち、O(log(n))のオーダーで検索することが可能になる。

[0077] 既に説明したように、ノード間の親子関係は、「子→親」関係の配列の他に、「親→子」関係の配列によっても表現できる。図13は、図7A～Cに示された幅優先のツリー型データ構造から作成された「親→子」表現に基づく親子関係の配列の説明図である。一つの親ノードに対して複数の子ノードが存在し得るので、親子関係の配列は、各ノードに対する子ノードの番号が格納されている領域を示すための配列Aggrと、子ノードの番号が格納されている配列P→Cの二つの配列により構成される。例えば、配列Aggrの先頭から2番目の要素Aggr[1]の値は"3"であり、これは、ノード[1]に対する子ノードの番号は、配列P→Cの要素P→C[3]以降に格納されていることを表している。これにより、ノード[0]、即ち、ルート・ノードに対する子ノードは、配列P→Cの先頭から3個の要素、P→C[0]の1、P→C[1]の2、及び、P→C[2]の3であることがわかる。

[0078] この「親→子」表現に基づく親子関係の配列の求め方を説明する。

(1)ノードの番号が配列P→Cの最大の添字(＝11)と一致する場合、このノードに属する子ノードは存在しない。したがって、処理は継続されない。

(2)同図に太字で表された親ノードの番号からAggr値を求める。このAggr値は、配

列P→Cの開始点を表す。

(3)太字で表された親ノード番号＋1に対応するAggr値を求める。このAggr値−1が配列P→Cの終了点である。

[0079] 例えば、ノード0の子ノードの開始点は、Aggr[0]、即ち、0であり、終了点は、Aggr[1]−1、即ち、3−1＝2である。したがって、ノード0の子ノードは、配列P→Cの0ー2番目の要素、即ち、1、2及び3である。

[0080] [ツリー型データ構造の表現形式の相互変換]

上述のように、ノードに連続番号を付与するための深さ優先モード及び幅優先モードは、それぞれ、固有の優れた性質を備えている。そこで、本発明の一実施例によるコンピュータシステムは、深さ優先に基づく「子→親」表現形式と、幅優先に基づく「子→親」表現形式と、「親→子」表現形式と、の間で相互に表現形式を変換する。図14は、本発明の一実施例による三つの表現形式の相互変換の関係を示す図である。

[0081] 図15は、本発明の一実施例によりコンピュータシステムによって実現されるツリー型データ構造の構築方法のフローチャートである。同図に示されるように、コンピュータシステム10は、ルート・ノードから始めて全てのノードに連続的に変化する整数を一意に割り当てるステップ1510と、ノード間に親子関係を定義するステップ1520と、を実行することにより、ツリー型データ構造を記憶装置上に構築する。

[0082] 好ましくは、前記全てのノードに整数を一意に割り当てるステップ1510は、

同じ世代のノードよりも子ノードを優先して番号を付与する深さ優先モードと、子ノードよりも同じ世代のノードを優先して番号を付与する幅優先モードのどちらのモードでノードに番号を付与するかを選択するステップ1511と、

前記深さ優先モードが選択された場合に、深さ優先でノードを検索し、検索された順にノードに番号を付与するステップ1512と、

前記幅優先モードが選択された場合に、幅優先でノードを検索し、検索された順にノードに番号を付与するステップ1513と、

を含む。これにより、深さ優先モードによるノード番号付与と幅優先モードによるノード番号付与を一つのシステムに併存させることができるので、状況に応じて適切な表現

形式を利用することが可能である。

[0083] また、好ましくは、前記ノード間に親子関係を定義するステップ1520は、

子ノードから親ノードへの関係を定義する子親表現モードと、親ノードから子ノードへの関係を定義する親子表現モードのどちらのモードで親子関係を定義するかを選択するステップ1521と、

前記子親表現モードが選択された場合に、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納するステップ1522と、

前記親子表現モードが選択された場合に、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号の配列を前記記憶装置に格納するステップ1523と、

を含む。これにより、「子→親」関係で表現されているノード間の親子関係を、状況に応じて、「親→子」関係で表現することが可能になる。「親→子」関係に基づく表現は、例えば、外部との情報交換の際に有利である。

[0084] このように、本発明の一実施例によれば、ツリー型データ構造の表現形式として、親子関係を表現するための「子→親」表現及び「親→子」表現、並びに、ノードに番号を付与するため深さ優先モード及び幅優先モードを選択的に利用可能である。以下では、異なる表現形式の相互変換の方法を説明する。

[0085] [深さ優先「子→親」表現から幅優先「子→親」表現への変換]

図16A、Bは、本発明の一実施例による深さ優先「子→親」表現(図16A)から、幅優先「子→親」表現(図16B)への変換の説明図である。図17は、この本発明の一実施例による深さ優先「子→親」表現から幅優先「子→親」表現への変換方法のフローチャートである。親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号をコンピュータシステム10の記憶装置、例えば、RAM14に格納することにより定義されている。図17に示されるように、コンピュータシステム10は、

同じ世代のノードよりも子ノードを優先してノードに番号を付与する深さ優先モードで表現されたツリー型データ構造の各ノードの世代を判定し、世代毎に属しているノ

ードの個数を計数するステップ1701と、

前記世代毎に属しているノードの個数に基づいて、子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで番号を付与する際に各世代において付与される番号を決定するステップ1702と、

前記判定されたノードの世代及び前記決定された各世代において付与される番号に基づいて、前記各ノードの番号を前記幅優先モードで付与される番号に変換する変換配列を作成するステップ1703と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記幅優先モードで付与される番号で表現された親子関係に変換するステップ1704と、

を実行する。これにより、深さ優先モードに基づく「子→親」表現形式から幅優先モードに基づく「子→親」表現形式への変換が可能になる。

[0086] 次に、上記ステップ1701ー1704をより詳細に説明する。

[0087] ステップ1701では、世代毎のノード個数を計数する。図18A乃至図22は、本発明の一実施例による深さ優先に基づくツリー型データ構造の世代毎に属しているノードの個数を計数する処理の説明図である。最初に、図18Aの手順0に示されるように、二つの変数配列を準備する。各ノードの世代を格納する配列depthは配列C→Pと同じサイズである。世代毎のノード個数を格納する配列depth−countは、ツリー構造の深さの段数以上の適当なサイズであり、0で初期化されている。図18Bの手順1では、先頭の要素（具体的には、ルート・ノード）から始めて、ノードの世代（深さ）を判定して配列depthに格納し、この要素の世代に属する要素の個数、即ち、配列depth−countの先頭の要素を1だけインクリメントする。ノード0は、世代が0であるため、depth[0]に0を設定し、depth−count[0]を0から1にインクリメントする。図中、対象となるノードの番号は、太字で示されている。図18Cの手順2では、ノード1に対応する親ノードの番号を配列C→Pから取得し、親ノードの世代を調べる。配列C→P[1]の要素は0であるため、depth[0]を参照すると、要素は0であるので（図中、イタリック体で示されている）、親ノードの世代は0であることがわかる。ノード1の世代の値は、親ノードの世代の値に1を加えた値であるから、親ノードの世代の値＋1＝1である。よって、配列depth[1]に世代の値1を設定し、配列depth−count[1]の要素を1だけイ

ンクリメントする。

[0088] 以下、図19A～Cの手順3～5、図20A～Cの手順6～8、図21A～Cの手順9～11、及び、図22の手順12の順番に、ノード2～ノード11に対し同様の処理を続ける。

[0089] 手順3:次のC→Pの要素は1なので、depth[1](親のdepthに該当する)を参照する。親のdepthは1であるので、自ノードのdepthは、1＋1＝＞2になる。そこで自ノードのdepth(＝2)をdepth[2]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0090] 手順4:次のC→Pの要素は2なので、depth[2](親のdepthに該当する)を参照する。親のdepthは2であるので、自ノードのdepthは、2＋1＝＞3になる。そこで自ノードのdepth(＝3)をdepth[3]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0091] 手順5:次のC→Pの要素は2なので、depth[2](親のdepthに該当する)を参照する。親のdepthは2であるので、自ノードのdepthは、2＋1＝＞3になる。そこで自ノードのdepth(＝3)をdepth[4]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0092] 手順6:次のC→Pの要素は1なので、depth[1](親のdepthに該当する)を参照する。親のdepthは1であるので、自ノードのdepthは、1＋1＝＞2になる。そこで自ノードのdepth(＝2)をdepth[5]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0093] 手順7:次のC→Pの要素は0なので、depth[0](親のdepthに該当する)を参照する。親のdepthは0であるので、自ノードのdepthは、0＋1＝＞1になる。そこで自ノードのdepth(＝1)をdepth[6]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0094] 手順8:次のC→Pの要素は6なので、depth[6](親のdepthに該当する)を参照する。親のdepthは1であるので、自ノードのdepthは、1＋1＝＞2になる。そこで自ノードのdepth(＝2)をdepth[7]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0095] 手順9:次のC→Pの要素は0なので、depth[0](親のdepthに該当する)を参照す

る。親のdepthは0であるので、自ノードのdepthは、0＋1＝＞1になる。そこで自ノードのdepth（＝1）をdepth[8]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0096] 手順10：次のC→Pの要素は8なので、depth[8]（親のdepthに該当する）を参照する。親のdepthは0であるので、自ノードのdepthは、1＋1＝＞2になる。そこで自ノードのdepth（＝2）をdepth[9]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0097] 手順11：次のC→Pの要素は9なので、depth[9]（親のdepthに該当する）を参照する。親のdepthは2であるので、自ノードのdepthは、2＋1＝＞3になる。そこで自ノードのdepth（＝3）をdepth[10]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0098] 手順12：次のC→Pの要素は9なので、depth[9]（親のdepthに該当する）を参照する。親のdepthは2であるので、自ノードのdepthは、2＋1＝＞3になる。そこで自ノードのdepth（＝3）をdepth[11]に格納する。最後に、depth−count[自ノードのdepth]をインクリメントする。

[0099] これにより、図23に示されるような配列depth及び配列depth−countが得られる。

[0100] 次に、ステップ1702において、世代毎のノードの個数が格納されている配列depth−countの要素を累計数にする（即ち、個数を累計する）。例えば、配列（1，3，4，4，0）の要素の値を累計数化すると、

1→1

3→1＋3＝4

4→4＋4＝8

4→8＋4＝12

0→12＋0＝12

により、（1，4，8，12，12）のようになる。この累計数から明らかであるように、世代0までのノード数は1個であり、世代1までのノード数は4個であり、世代2までのノード数は8個であり、世代3までのノード数は12個であり、世代4までのノード数は12個である。ここから、世代順にノードを並べた場合、世代0の先頭ノードは全体で0番目であ

り、世代1の先頭ノードは全体で1番目のノードであり、世代2の先頭ノードは全体で4番目のノードであり、世代3の先頭ノードは全体で8番目のノードであり、世代4の先頭ノードは全体で12番目のノードであることがわかる。このように、世代毎のノード個数の配列depth−countの要素を累計数化することにより、世代順にノードを並べた場合の各世代の先頭ノードが全体で何番目のノードであるかを示す配列depth−aggrが得られる。尚、好ましい一実施例では、配列depth−aggrは、(1, 4, 8, 12, 12)そのままではなく、先頭に0を詰め、要素を一つずつ後へずらすことにより得られる配列(0, 1, 4, 8, 12)によって与えられる。この配列depth−aggrは、幅優先モードで番号を付与する際に各世代において付与される番号を表している。

[0101] ステップ1703では変換配列を作成する。図24A乃至図28は、本発明の一実施例によるノードの番号を幅優先モードで付与される番号に変換する変換配列を作成する処理の説明図である。最初に、図24Aの手順0に示されるように、配列C→Pと同じサイズの整数配列である番号変換定義配列「No. の変換定義」の領域を確保する。次に、図24Bの手順1において、ノード0の世代、即ち、depth[0]を取り出し、その値0が指定する配列depth−aggrの要素depth−aggr[0]の要素0を取り出す。この値0は、上述のように、世代0において付与される番号を表している。ノード0に対応する配列「No. の変換定義」の要素にこの値0を設定することにより、深さ優先モードで番号が付与されたノード0は、幅優先モードで番号を付与した場合に、ノード0に変換されることがわかる。

[0102] 図24Cの手順2では、ノード1の番号変換定義を行う。そのため、ノード1の世代を表すdepth[1]を取り出し、その値1が指定するdepth−aggr[1]の値1を取り出す。この値1を配列「No. の変換定義」[1]に設定すると共に、depth−aggr[1]の値を1だけ増加させる。要素が取り出されたdepth−aggr[1]の要素を1だけインクリメントすることにより、次に、世代1のノードが選択された場合、そのノードの変換後の番号は、ノード1の変換後の番号1よりも1だけ大きい番号、即ち、2になる。

[0103] 以下、図25A−Cの手順3−5、図26A−Cの手順6−8、図27A−Cの手順9−11、及び、図28の手順12の順番に、ノード2−ノード11に対し同様の処理を続ける。

[0104] 例えば、手順3では、depth[2]を取り出し、その値が指定するdepth−aggrの要素

を取り出す。取り出したdepth−aggrの該当する要素は、「No. の変換定義」に格納すると共に1だけ増加させて、depth−aggrに格納する。手順4ー12では、それぞれ、depth[3]ーdepth[11]を取り出し、その値が指定するdepth−aggrの要素を取り出す。取り出したdepth−aggrの該当する要素は、「No. の変換定義」に格納すると共に1だけ増加させて、depth−aggrに格納する。

[0105] これにより、図28に示されるような最終的な配列「No. の変換定義」が得られる。

[0106] ステップ1704では、各ノードの親子関係を、変換配列を使用して幅優先モードで付与される番号で表現された親子関係に変換する。図29は、本発明の一実施例による深さ優先モードに基づく「子→親」表現形式の親子関係を幅優先モードに基づく「子→親」表現形式の親子関係に変換する処理の説明図である。例えば、深さ優先モードに基づく「子→親」表現形式で、子ノードCと親ノードPが関連付けられている場合、上記の最終的な番号変換定義配列「No. の変換定義」によって、ノード番号Cがノード番号C'に変換され、ノード番号Pがノード番号P'に変換されるならば、幅優先モードに基づく「子→親」表現形式では、子ノードC'と親ノードP'が関連付けられることになる。変換前の全ての子ノードCに対して、変換後の子ノードの番号C'と変換後の親ノードの番号P'が得られたならば、格納位置がC'で表され、格納値がP'で表される親子関係の配列C'→P'が完成する。

[0107] 図29の例では、最初に、格納位置(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11)は、「No. の変換定義」配列を用いて、格納位置(0, 1, 4, 8, 9, 5, 2, 6, 3, 7, 10, 11)に変換される。これにより、同図に示されるように、格納位置の変換後の配列C→Pは、

格納位置の変換後の配列C→P[0]には、深さ優先による配列C→P[0]の値−1がセットされ、

格納位置の変換後の配列C→P[1]には、深さ優先による配列C→P[1]の値0がセットされ、

格納位置の変換後の配列C→P[2]には、深さ優先による配列C→P[6]の値0がセットされ、

格納位置の変換後の配列C→P[3]には、深さ優先による配列C→P[8]の値0がセ

ットされ、

格納位置の変換後の配列C→P[4]には、深さ優先による配列C→P[2]の値1がセットされ、

格納位置の変換後の配列C→P[5]には、深さ優先による配列C→P[5]の値1がセットされ、

格納位置の変換後の配列C→P[6]には、深さ優先による配列C→P[7]の値6がセットされ、

格納位置の変換後の配列C→P[7]には、深さ優先による配列C→P[9]の値8がセットされ、

格納位置の変換後の配列C→P[8]には、深さ優先による配列C→P[3]の値2がセットされ、

格納位置の変換後の配列C→P[9]には、深さ優先による配列C→P[4]の値2がセットされ、

格納位置の変換後の配列C→P[10]には、深さ優先による配列C→P[10]の値9がセットされ、

格納位置の変換後の配列C→P[11]には、深さ優先による配列C→P[11]の値9がセットされる。

[0108] 次に、格納位置の変換後の配列C→Pの各要素の値を「No. の変換定義」配列を用いて変換することにより、格納値の変換後のC→P配列が得られる。

[0109] 図29の例では、最初に格納位置の変換、即ち、子ノード番号の変換を行い、その後、格納値の変換、即ち、親ノード番号の変換を行っているが、格納値の変換を先に行った後に格納位置の変換を行ってもよく、或いは、格納位置の変換と格納値の変換を同時に行ってもよい。尚、ルート・ノードの親ノードを表すノード番号”−1”は、変換する必要がない。

[0110] 以上の処理により、図16A、Bに示されるような深さ優先「子→親」表現から幅優先「子→親」表現への変換が行われる。

[0111] [幅優先「子→親」表現から深さ優先「子→親」表現への高速変換]

図30A、Bは、それぞれ、本発明の一実施例による幅優先「子→親」表現（図30A）

から深さ優先「子→親」表現（図30B）への変換の説明図である。図31は、この本発明の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法のフローチャートである。親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号をコンピュータシステム10の記憶装置、例えば、RAM14に格納することにより定義されている。図31に示されるように、コンピュータシステム10は、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードの子孫の数を計数するステップ3101と、

親ノードに付与されるべき番号に、当該ノードと同一の親ノードから派生した兄弟ノードのうち当該ノードよりも前に番号が付与されている兄ノードの数及び当該兄ノードの子孫の数を加算することにより、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成するステップ3102と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換するステップ3103と、

を実行する。これにより、幅優先モードに基づく「子→親」表現形式から深さ優先モードに基づく「子→親」表現形式への高速変換が可能になる。

[0112] 次に、上記ステップ3101ー3103をより詳細に説明する。

[0113] ステップ3101では、ノードの子孫の数を計数する。図32乃至図33は、本発明の一実施例による幅優先に基づくツリー型データ構造の各ノードの子孫の個数を計数する処理の説明図である。最初に、図32に示されるように、配列C→Pと同じサイズのノード数配列を作成し、ノード数配列の各要素を1で初期化する。

[0114] 次に、図33に示されるように、世代を遡る順番に（即ち、ルート・ノードへ向かって）、配列C→Pの要素の値を使ってノード数配列にアクセスし、配列C→Pの子ノードに対応するノード数配列の要素の値を、配列C→Pの親ノードに対応するノード数配列の要素の値に加算する。これにより、親ノードに対応するノード数配列の要素には、その親ノードの子孫ノードの個数が加算される。例えば、図33の手順1では、配列C→P[11]の値から、子ノード11の親ノードはノード7であることがわかる。そこで、子ノ

ード11に対応するノード数配列の要素、即ち、ノード数配列[11]の値を、親ノード7に対応するノード数配列の要素、即ち、ノード数配列[7]の値に加算する。これにより、子ノード11の子孫ノードの数が親ノード7の子孫ノードの数に含まれる。手順2乃至手順11に従って同様の処理を行うと、図33に示されるような、ノード数の加算により得られたノード数配列が作成される。尚、本例では、配列C→Pの先頭の要素である配列C→P[0]は、ルート・ノードに対応しているので、ノード数の加算は、配列の要素C→P[1]に対する手順11の処理で終了する。既に説明したように、ルート・ノードを配列C→Pから除いても構わない。最終的なノード数配列の要素の値は、自ノードもカウントされているので、例えば、ノード0の子孫ノードの個数は、(12−1)＝11個である。

[0115] ステップ3102では、幅優先モードで番号が付与されたノード毎に、親ノードの番号に、兄ノードの数と兄ノードの子孫の数を加算することにより、当該ノードの幅優先モードによるノードの番号を深さ優先モードの番号に変換する変換配列を作成する。上述のように、幅優先モードは、子ノードよりも同じ世代のノードを優先してノードに番号を付与するものであり、深さ優先モードは、同じ世代のノードよりも子ノードを優先してノードの番号を付与するものである。そのため、ある親ノードに唯一の子ノードが存在する場合、深さ優先モードでは、この子ノードには、親ノードの次の番号が付与される。一方、ある親ノードに複数の子ノードが存在する場合、ある子ノードの番号は、その子ノードの兄ノード及び兄ノードの全ての子孫ノードに番号が付与された後に付与される。ある兄ノードの全ての子孫ノードの個数は、ノード数配列を参照することにより得ることができる。

[0116] 図34は、本発明の一実施例による幅優先モードの番号から深さ優先モードの番号への変換配列を作成する処理の説明図である。この処理は、ルート・ノード0から順番に進められる。手順1では、配列C→Pにおいて、ルート・ノード0の参照先、即ち、親ノードは存在しないので、ルート・ノードの番号は0のままである。そこで、ルート・ノードに対するノード数配列の要素の値は0に書き換えられる。

[0117] 次に、手順1では、配列C→Pの要素の値が0であるノード、即ち、ノード0を親ノードとしてもつノードが処理の対象となる。本例では、ノード1、ノード2及びノード3は、共

通の親ノード0から派生した子ノードであるので、兄弟ノードと呼ばれる。幅優先モードでは、兄弟ノードの中に長子から末子へ向かって、ノード1、ノード2、ノード3の順序が付けられている。長子であるノード1は、兄ノードが存在しないので、親ノードであるノード0の次の番号を付与すればよい。したがって、ノード1に対するノード数配列の要素の値は、5から1に書き換えられる。次に、ノード2は、兄ノードであるノード1が存在するので、ノード1以下のノードの個数5を、親ノードのノード番号0に加算し、更に、自ノードの個数分の1を加算して得られるノード番号6に変換される。末子のノード3は、兄ノード1と兄ノード2が存在するので、兄ノード1以下のノードの個数5と、兄ノード2以下のノードの個数2と、親ノードのノード番号0と、自ノードの個数1と、を加算することにより得られるノード番号8に変換される。

[0118] 手順2から手順7まで、同一の親から派生した兄弟ノードの単位で手順1と同様の処理を繰り返すことにより、ノード数配列は変換配列に書き換えられる。勿論、ノード数配列とは別に変換配列を作成しても構わない。

[0119] ステップ3103では、各ノードの親子関係を、変換配列を使用して深さ優先モードで付与される番号で表現された親子関係に変換する。図35は、本発明の一実施例による幅優先モードに基づく「子→親」表現形式の親子関係を深さ優先モードに基づく「子→親」表現形式の親子関係に変換する処理の説明図である。この処理は、図29に関して説明した処理と同様の処理である。例えば、幅優先モードに基づく「子→親」表現形式で、子ノードCと親ノードPが関連付けられている場合、上記の変換配列(「No. の変換定義」)によって、ノード番号Cがノード番号C'に変換され、ノード番号Pがノード番号P'に変換されるならば、深さ優先モードに基づく「子→親」表現形式では、子ノードC'と親ノードP'が関連付けられることになる。変換前の全ての子ノードCに対して、変換後の子ノードの番号C'と変換後の親ノードの番号P'が得られたならば、格納位置がC'で表され、格納値がP'で表される親子関係の配列C'→P'が完成する。

[0120] 図35の例では、最初に格納位置の変換、即ち、子ノード番号の変換を行い、その後、格納値の変換、即ち、親ノード番号の変換を行っているが、格納値の変換を先に行った後に格納位置の変換を行ってもよく、或いは、格納位置の変換と格納値の変

換を同時に行ってもよい。尚、ルート・ノードの親ノードを表すノード番号”−1”は、変換する必要がない。

[0121] 以上の処理により、図30A、Bに示されるような幅優先「子→親」表現から深さ優先「子→親」表現への高速変換が行われる。

[0122] ［幅優先「子→親」表現から深さ優先「子→親」表現への変換］

本発明の他の一実施例によれば、幅優先「子→親」表現から深さ優先「子→親」表現への変換は、図30A乃至図35に関して説明した高速変換方法の他に、検索を利用した変換方法によっても実現できる。以下では、図30A、Bに示された例を用いて、この検索を利用した変換方法を説明する。

[0123] 幅優先「子→親」表現による親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号をコンピュータシステム10の記憶装置、例えば、RAM14に格納することにより定義されている。コンピュータシステム10は、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードを深さ優先で検索し、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成するステップと、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換するステップと、

を実行する。

[0124] 図36乃至図43Bは、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図である。幅優先モードに基づく「子→親」表現形式から深さ優先モードに基づく「子→親」表現形式への検索に基づく変換において、「深さ優先」検索は、例えば、スタックを使用して番号変換配列を作成することにより実現される。

[0125] 変換配列を作成するステップの第1段階では、図36に示されるような変数を作成する。変換配列は、配列C→Pと同じサイズの整数配列であり、−1で初期化される。変数CURRENT_NO及び変数STACK_POSは0で初期化される。配列STACK

は適当な長さの整数配列である。

[0126] 変換配列を作成するステップの第2段階では、図37A乃至図43Bの手順1ー手順20に示されるように、トレースを行いながら変換配列を作成する。

[0127] 手順1:(1)C→Pの要素で−1を含む場所(ルート・ノードを示す)を探し、添字0=0の場所を見つける。(2)添字(=0)をSTACKに格納し、同時に、STACK__POSをインクリメントする。(3)添字(=0)と同じ位置の変換配列の要素にCURRENT__NOの値を格納し、CURRNT__NOをインクリメントする。

[0128] 手順2:(1)STACK__POS−1のスタック格納値(=0)を指す、未使用で最小のC→Pの位置を検索する。未使用か否かは、変換配列の要素が1であるかどうかで判定できる。未使用かつ最小の添字は(=1)に存在することが分かるので、その位置にポインタ(矢印マーク)を設定する。(2)この添字(=1)をSTACKに格納し、STACK__POSをインクリメントする。(3)添字(=1)と同じ位置の変換配列の要素にCURRENT__NOの値を格納し、CURRNT__NOをインクリメントする。

[0129] 手順3:(1)STACK__POS−1のスタック格納値(=1)を指す、未使用で最小のC→Pの位置を検索する。未使用か否かは、変換配列の要素が1であるかどうかで判定できる。未使用かつ最小の添字は(=4)に存在することが分かるので、その位置にポインタ(矢印マーク)を設定する。(2)この添字(=4)をSTACKに格納し、STACK__POSをインクリメントする。(3)添字(=4)と同じ位置の変換配列の要素にCURRENT__NOの値を格納し、CURRNT__NOをインクリメントする。

[0130] 手順4:(1)STACK__POS−1のスタック格納値(=4)を指す、未使用で最小のC→Pの位置を検索する。未使用か否かは、変換配列の要素が1であるかどうかで判定できる。未使用かつ最小の添字は(=8)に存在することが分かるので、その位置にポインタ(矢印マーク)を設定する。(2)この添字(=8)をSTACKに格納し、STACK__POSをインクリメントする。(3)添字(=8)と同じ位置の変換配列の要素にCURRENT__NOの値を格納し、CURRNT__NOをインクリメントする。

[0131] 手順5:(1)STACK__POS−1のスタック格納値(=8)を指す、未使用で最小のC→Pの位置を検索する。未使用か否かは、変換配列の要素が1であるかどうかで判定できる。未使用かつ最小の添字は存在しないことがわかるので、STACK__POSを

デクリメントする。

[0132] 手順6：(1)STACK__POS−1のスタック格納値(＝4)を指す、未使用で最小のC→Pの位置を検索する。未使用か否かは、変換配列の要素が1であるかどうかで判定できる。未使用かつ最小の添字は(＝9)に存在することが分かるので、その位置にポインタ(矢印マーク)を設定する。(2)この添字(＝9)をSTACKに格納し、STACK__POSをインクリメントする。(3)添字(＝9)と同じ位置の変換配列の要素にCURRENT__NOの値を格納し、CURRNT__NOをインクリメントする。

[0133] 以下同様に手順7ー手順20を実行する。手順20の終了時点で、CURRENT__NOが変換配列のサイズ(＝12)に達し、変換配列が完成したことがわかる。ここで、変換配列作成処理は終了する。

[0134] この検索に基づく変換方法によって作成された変換配列は、先の高速変換方法の例で作成された図34の変換配列と同じであることがわかる。

[0135] 次に、図35を参照して説明した本発明の一実施例による幅優先モードに基づく「子→親」表現形式の親子関係を深さ優先モードに基づく「子→親」表現形式の親子関係に変換する処理と全く同様の処理によって、親子関係の表現を変換する。

[0136] [「子→親」表現から「親→子」表現への変換]

次に、本発明の一実施例による子ノードに親ノードを対応付ける「子→親」関係から親ノードに子ノードを対応付ける「親→子」関係への変換方法を説明する。

[0137] 図44は、本発明の一実施例による「子→親」表現から「親→子」表現への変換方法のフローチャートである。親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を第1の配列の要素としてコンピュータシステム10の記憶装置、例えば、RAM14に格納することにより定義されている。コンピュータシステム10は、

各ノードに関して、当該ノードに付与された番号が前記第1の配列の要素として出現する回数を計数するステップ4401と、

前記各ノードに関して当該ノードに対応する子ノードに付与された番号を格納するため、前記計数された回数に応じた個数の連続領域を前記記憶領域に第2の配列として確保するステップ4402と、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する子ノードの番号を順次に格納するステップ4403と、

を実行する。これにより、親子関係は、「子→親」表現形式から「親→子」表現形式に変換される。即ち、変換後の親子関係は、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号を前記第2の配列の要素として前記記憶装置に格納することにより定義される。

[0138] この変換方法では、深さ優先又は幅優先の性質はそのまま保存されるので、深さ優先モードに基づく「子→親」表現は、深さ優先モードに基づく「親→子」表現に変換され、幅優先モードに基づく「子→親」表現は、幅優先モードに基づく「親→子」表現に変換される。以下では、深さ優先モードに基づく「子→親」表現から深さ優先モードに基づく「親→子」表現への変換の例を説明する。図45A～Cは、それぞれ、深さ優先モードに基づくツリー型データ構造の一例の説明図であり、図45Aはツリー型データ構造の全体を示し、図45Bは「子→親」表現形式による親子関係を示し、図45Cは「親→子」表現形式による親子関係を示している。本実施例では、図45Bのような表現形式を、図45Cのような表現形式へ変換する。

[0139] 図46A乃至図47Cは、本発明の一実施例による深さ優先モードに基づく「子→親」表現から深さ優先モードに基づく「親→子」表現への変換方法の説明図である。

[0140] 図46Aの手順1では、最初に、変換後の「親→子」表現形式の親子関係を格納するための領域を確保し、初期化する。既に説明したように、「親→子」表現形式の場合、配列Aggrと配列P→Cを用意する。配列Aggrは、各ノードに対する子ノードの番号が格納されている領域を示すための配列であり、配列P→Cは子ノードの番号を格納するための配列である。配列Aggrのサイズは、(「子→親」表現形式の)配列C→Pと同じサイズであり、配列Aggrは0で初期化する。配列P→Cのサイズは、配列Aggrよりも要素1個分小さいサイズで足りる。配列P→Cは、特に初期化する必要はないが、図46Aでは、理解しやすいように−1で初期化されている。

[0141] 図46Bの手順2では、配列C→Pの各要素が指定する配列Aggrの要素を1ずつインクリメントする。配列C→Pの要素の値は、親ノードの番号を表しているので、カウン

トアップされた配列Aggrの各要素は、配列Aggrの添字の番号と一致するノードの子ノードの個数を表す。

[0142] 図46Cの手順3では、カウントアップされたAggrの要素の値を累計数に変換する。これにより、各ノードに対する子ノードの番号が格納されている領域を示すための配列Aggrが完成する。尚、本例では、累計数にするときに、1つずつ後へシフトさせている。

[0143] 図47Aの手順4では、配列C→Pから配列P→Cへノード番号を転送する。配列C→Pの先頭の要素は、ルート・ノード0についての情報であり、ルート・ノード0には親ノードが定義されていないので(本例では、配列C→Pの要素が負の値−1であるので何もしない)、実際の処理は、ノード1に対応する配列C→Pの添字1から始まる。配列C→Pの添字1の要素は、ノード1の親ノードがノード0であることを示している。そこで、配列Aggrの中でノード0に対する要素、即ち、添字1に対応する要素を参照すると、値0が格納されているので、配列P→Cの中でこの値0で指定される格納場所の値、即ち、配列P→Cの添字0の要素に、配列C→Pの添字1に対応するノード1のノード番号を設定する。これにより、ノード0の子ノードとしてノード1が設定される。このとき、配列Aggrの中でノード0に対する要素の値をインクリメントする。これにより、ノード0の別の子ノードが検出されたとき、その子ノードの番号は、配列P→Cの添字1の要素の値として設定されることになる。本例では、配列C→Pを参照すると、ノード6の親ノードがノード0であるため、配列P→Cの添字1の要素の値として、このノード6のノード番号6が設定されている。

[0144] 次に、図47Bの手順5では、配列Aggrを手順3の終了時の状態に戻す。これは、例えば、図47Bに示されるように、配列Aggrの要素を1個分ずつ後へシフトし、先頭に0を詰めることにより実現できる。或いは、手順3の終了時の配列Aggrを別途保存しておいてもよい。或いは、配列Aggrの先頭アドレスを一つ前に付け替てもよい。

[0145] 図47Cには、変換によって得られた配列Aggrと配列P→Cが示されている。これらの配列は、図10に示した深さ優先「親→子」関係に基づく親子関係の配列と同じであるので、これ以上の説明は加えない。

[0146] この変換方法では、既に説明したように、深さ優先又は幅優先の性質はそのまま保

存される。したがって、本発明の一実施例による変換方法は、幅優先モードに基づく「子→親」表現から幅優先モードに基づく「親→子」表現への変換にも適合する。

[0147] [「親→子」表現から「子→親」表現への変換]

次に、本発明の一実施例による親ノードに子ノードを対応付ける「親→子」関係から子ノードに親ノードを対応付ける「子→親」関係への変換方法を説明する。

[0148] 図48は、本発明の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法のフローチャートである。親子関係は、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号を第1の配列の要素としてコンピュータシステム10の記憶装置、例えば、RAM14に格納することにより定義されている。コンピュータシステム10は、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を格納するため、前記記憶装置に第2の配列を確保するステップ4801と、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する親ノードの番号を順次に格納するステップ4802と、

を実行する。これにより、親子関係は、「親→子」表現形式から「子→親」表現形式に変換される。即ち、変換後の親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記第2の配列の要素として前記記憶装置に格納することにより定義される。

[0149] この変換方法では、深さ優先又は幅優先の性質はそのまま保存されるので、深さ優先モードに基づく「親→子」表現は、深さ優先モードに基づく「子→親」表現に変換され、幅優先モードに基づく「親→子」表現は、幅優先モードに基づく「子→親」表現に変換される。以下では、深さ優先モードに基づく「親→子」表現から深さ優先モードに基づく「子→親」表現への変換の例を説明する。本例では、図45A－Cに示された深さ優先モードに基づくツリー型データ構造に関して、図45Cに示された「親→子」表現形式による親子関係を、図45Bに示された「子→親」表現形式による親子関係へ変換する。

[0150] 図49A－Cは、それぞれ、本発明の一実施例による深さ優先モードに基づく「親→

子」表現から深さ優先モードに基づく「子→親」表現への変換方法の説明図である。

[0151] 最初に、図49Aの手順1に示されるように、配列C→Pの領域を確保し、−1で初期化する。

[0152] 次に、図49Bの手順2−1及び図49Cの手順2−2に示されるように、配列Aggrと配列P→Cから「親→子」関係を読み出し、対応する配列C→Pの子ノード番号を埋める。例えば、配列Aggrの添字0の要素(＝0)は、配列P→Cにおいて親ノード0の子ノードが格納されている領域の先頭を示し、配列Aggrの添字1の要素(＝3)は、配列P→Cにおいて親ノード1の子ノードが格納されている領域の先頭を示しているので、親ノード0の子ノードのノード番号は、配列P→Cの添字0から添字2までの領域に格納されていることがわかる。この領域には、子ノード番号として、1、6及び8が順番に格納されているので、配列C→Pの添字1、添字6及び添字8の要素として、親ノード0のノード番号0を設定する。これにより、配列C→Pの中で、ノード番号0のノードを親ノードとする子ノードの領域が埋められる。この手続を配列Aggrの添字の順に実行することにより、図49Dに示される最終結果が得られる。

[0153] この変換方法では、既に説明したように、深さ優先又は幅優先の性質はそのまま保存される。したがって、本発明の一実施例による変換方法は、幅優先モードに基づく「親→子」表現から幅優先モードに基づく「子→親」表現への変換にも適合する。

[0154] [情報処理装置]

図50は、本発明の一実施例によるツリー型データ構造を構築する情報処理装置5000のブロック図である。情報処理装置5000は、ツリー型データ構造を表現するデータを記憶する記憶部5001と、ルート・ノードを含むノードに固有のノード識別子を付与するノード定義部5002と、前記ルート・ノード以外のノードである非ルート・ノードの各々に付与されたノード識別子に、前記非ルート・ノードの各々の親ノードに付与されたノード識別子を関連付ける親子関係定義部5003と、を含む。

[0155] 好ましくは、ノード定義部5002は、ノード識別子として数値を用い、より好ましくは、ノード識別子として連続する整数を用いる。

[0156] また、親子関係定義部5003は、非ルート・ノードの各々に付与されたノード識別子と、関連付けられた親ノードに付与されたノード識別子と、の組の配列を記憶部5001

に格納する。

[0157] 本発明は、以上の実施の形態に限定されることなく、特許請求の範囲に記載された発明の範囲内で、種々の変更が可能であり、それらも本発明の範囲内に包含されるものであることは言うまでもない。

図面の簡単な説明

[0158] [図1]図1は、本発明の実施の形態にかかるツリー型データ構造を取り扱うコンピュータシステムのブロックダイヤグラムである。

[図2]図2A，Bは、ツリー形式データの一例であるPOSデータの説明図であり、図2Aは、このツリー形式データのデータ構造（即ち、トポロジー）及びデータ値を視覚的に表現した例であり、図2Bは、同じツリー形式データをXML形式で表現した例である。

[図3]図3A－Cは、それぞれ、アークリストを用いたツリー型データ構造の表現形式の一例の説明図である。

[図4]図4A－Cは、それぞれ、本発明の一実施例による「子→親」関係に基づくツリー型データ構造の表現方法の説明図である。

[図5]図5は、本発明の一実施例によるツリー型データ構造を記憶装置上に構築する方法のフローチャートである。

[図6]図6A－Cは、それぞれ、本発明の一実施例によりID形式のツリー構造型データを整数連番形式のツリー構造型データへ変換する処理の説明図である。

[図7]図7A－Cは、それぞれ、本発明の他の一実施例によりID形式のツリー構造型データを整数連番形式のツリー構造型データへ変換する処理の説明図である。

[図8]図8は、本発明の一実施例による深さ優先に基づくノード定義処理のフローチャートである。

[図9]図9は、本発明の一実施例により作成された「子→親」表現に基づく親子関係の配列の説明図である。

[図10]図10は、図6A－Cに示された深さ優先のツリー型データ構造から作成された「親→子」表現に基づく親子関係の配列の説明図である。

[図11]図11は、本発明の一実施例による幅優先に基づくノード定義処理のフローチャートである。

[図12]図12は、本発明の一実施例により作成された「子→親」表現に基づく親子関係の配列の説明図である。

[図13]図13は、図7A～Cに示された深さ優先のツリー型データ構造から作成された「親→子」表現に基づく親子関係の配列の説明図である。

[図14]図14は、本発明の一実施例による三つの表現形式の相互変換の関係を示す図である。

[図15]図15は、本発明の一実施例によるコンピュータシステムによって実現されるツリー型データ構造の構築方法のフローチャートである。

[図16]図16A、Bは、それぞれ、本発明の一実施例による深さ優先「子→親」表現から幅優先「子→親」表現への変換の説明図である。

[図17]図17は、本発明の一実施例による深さ優先「子→親」表現から幅優先「子→親」表現への変換方法のフローチャートである。

[図18]図18A～Cは、それぞれ、本発明の一実施例による深さ優先に基づくツリー型データ構造の世代毎に属しているノードの個数を計数する処理の説明図である。

[図19]図19A～Cは、それぞれ、本発明の一実施例による深さ優先に基づくツリー型データ構造の世代毎に属しているノードの個数を計数する処理（手順0～2）の説明図である。

[図20]図20A～Cは、それぞれ、本発明の一実施例による深さ優先に基づくツリー型データ構造の世代毎に属しているノードの個数を計数する処理（手順3～5）の説明図である。

[図21]図21A～Cは、それぞれ、本発明の一実施例による深さ優先に基づくツリー型データ構造の世代毎に属しているノードの個数を計数する処理（手順6～8）の説明図である。

[図22]図22は、本発明の一実施例による深さ優先に基づくツリー型データ構造の世代毎に属しているノードの個数を計数する処理（手順9～11）の説明図である。

[図23]図23は、本発明の一実施例による深さ優先に基づくツリー型データ構造の世代毎に属しているノードの個数を計数する処理（手順12）の説明図である。

[図24]図24A～Cは、それぞれ、本発明の一実施例によるノードの番号を幅優先モ

ードで付与される番号に変換する変換配列を作成する処理（手順0ー2）の説明図である。

[図25]図25AーCは、それぞれ、本発明の一実施例によるノードの番号を幅優先モードで付与される番号に変換する変換配列を作成する処理（手順3ー5）の説明図である。

[図26]図26AーCは、それぞれ、本発明の一実施例によるノードの番号を幅優先モードで付与される番号に変換する変換配列を作成する処理（手順6ー8）の説明図である。

[図27]図27AーCは、それぞれ、本発明の一実施例によるノードの番号を幅優先モードで付与される番号に変換する変換配列を作成する処理（手順9ー11）の説明図である。

[図28]図28は、本発明の一実施例によるノードの番号を幅優先モードで付与される番号に変換する変換配列を作成する処理（手順12）の説明図である。

[図29]図29は、本発明の一実施例による深さ優先に基づくノードの親子関係を幅優先に基づくノードの親子関係に変換する処理の説明図である。

[図30]図30A、Bは、それぞれ、本発明の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換の説明図である。

[図31]図31は、本発明の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法のフローチャートである。

[図32]図32は、本発明の一実施例による幅優先に基づくツリー型データ構造の各ノードの子孫の個数を計数する処理の説明図（その1）である

[図33]図33は、本発明の一実施例による幅優先に基づくツリー型データ構造の各ノードの子孫の個数を計数する処理の説明図（その2）である

[図34]図34は、本発明の一実施例による幅優先モードの番号から深さ優先モードの番号への変換配列を作成する処理の説明図である。

[図35]図35は、本発明の一実施例による幅優先に基づくノードの親子関係を深さ優先に基づくノードの親子関係に変換する処理の説明図である。

[図36]図36は、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子

→親」表現への変換方法の説明図（その1）である。

[図37]図37AーCは、それぞれ、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図（その2）である。

[図38]図38AーCは、それぞれ、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図（その3）である。

[図39]図39AーCは、それぞれ、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図（その4）である。

[図40]図40AーCは、それぞれ、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図（その5）である。

[図41]図41AーCは、それぞれ、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図（その6）である。

[図42]図42AーCは、それぞれ、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図（その7）である。

[図43]図43A、Bは、それぞれ、本発明の他の一実施例による幅優先「子→親」表現から深さ優先「子→親」表現への変換方法の説明図（その8）である。

[図44]図44は、本発明の一実施例による「子→親」表現から「親→子」表現への変換方法のフローチャートである。

[図45]図45AーCは、それぞれ、深さ優先モードに基づくツリー型データ構造の一例の説明図である。

[図46]図46AーCは、それぞれ、本発明の一実施例による深さ優先モードに基づく「子→親」表現から深さ優先モードに基づく「親→子」表現への変換方法の説明図（その1）である。

[図47]図47AーCは、それぞれ、本発明の一実施例による深さ優先モードに基づく「子→親」表現から深さ優先モードに基づく「親→子」表現への変換方法の説明図（その2）である。

[図48]図48は、本発明の一実施例による「親→子」表現から「子→親」表現への変換方法のフローチャートである。

[図49]図49AーDは、それぞれ、本発明の一実施例による深さ優先モードに基づく「

親→子」表現から深さ優先モードに基づく「子→親」表現への変換方法の説明図である。

[図50]図50は、本発明の一実施例によるツリー型データ構造を記憶装置上に構築する情報処理装置のブロック図である。

符号の説明

[0159]

10	コンピュータシステム
12	CPU
14	RAM
16	ROM
18	固定記憶装置
20	CD−ROMドライバ
22	I／F
24	入力装置
26	表示装置
5000	情報処理装置
5001	記憶部
5002	ノード定義部
5003	親子関係定義部

請求の範囲

[1] ツリー型データ構造を構成するノード間の親子関係を記憶装置上で表現する方法であって、

ルート・ノード以外のノードである非ルート・ノードの各々に対して、該非ルート・ノードの親ノードを関連付けることにより前記ノード間の親子関係を表現する、

方法。

[2] ツリー型データ構造を記憶装置上に構築する方法であって、

ルート・ノードを含むノードに固有のノード識別子を付与するノード定義ステップと、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与されたノード識別子に、前記非ルート・ノードの各々の親ノードに付与されたノード識別子を関連付ける親子関係定義ステップと、

を含む方法。

[3] 前記ノード定義ステップは前記ノード識別子として数値を用いる、請求項2に記載の方法。

[4] 前記ノード定義ステップは前記ノード識別子として連続する整数を用いる、請求項2に記載の方法。

[5] 前記ノード間の親子関係は、前記非ルート・ノードの各々に関連付けられた前記親ノードの配列によって表現される、請求項1に記載の方法。

[6] 前記親子関係定義ステップは、前記非ルート・ノードの各々に付与されたノード識別子に関連付けられた前記親ノードに付与されたノード識別子の配列を前記記憶装置に格納する、請求項2に記載の方法。

[7] ツリー型データ構造を記憶装置上に構築する方法であって、

同じ世代のノードよりも子ノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義ステップと、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義ステップと、

を含む方法。

[8] 前記ノード定義ステップは、

最初にルート・ノードに番号を付与するステップと、

既に番号が付与されたあるノードに唯一の子ノードが存在する場合には、当該子ノードに当該あるノードに付与された前記番号の次の番号を付与するステップと、

既に番号が付与されたあるノードに複数の子ノードが存在する場合には、当該複数の子ノードの間の兄弟関係に従って、弟ノードは直上の兄ノードの全ての子孫ノードに番号が付与された後に次の番号が付与されるように、一番上の兄ノードから一番下の弟ノードまで番号を付与するステップと、

を含む、請求項7に記載の方法。

[9] ツリー型データ構造を記憶装置上に構築する方法であって、

子ノードよりも同じ世代のノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義ステップと、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義ステップと、

を含む方法。

[10] 前記ノード定義ステップは、

各ノードが前記ルート・ノードから何世代目のノードであるか、及び、各世代に含まれるノード数を算出するステップと、

最初に前記ルート・ノードに番号を付与するステップと、

ある世代に含まれる全てのノードに番号が付与されたならば、当該ある世代の次の世代にノードが存在しなくなるまで、当該次の世代に含まれる全てのノードに対して、親ノードが異なる場合には、当該親ノードに番号が付与された順番に当該ノードに番号を付与し、当該親ノードが同一である場合には、当該親ノードから派生した複数の子ノードの間に兄弟関係を定義し、一番上の兄ノードから一番下の弟ノードまで直前に付与された番号の次の番号から連続的に変化する固有の整数を順に付与するステップと、

を含む、請求項9に記載の方法。

[11] 前記配列から、あるノードに付与された整数以上の値が格納されている連続領域を抽出することにより、前記あるノードの全ての子孫ノードを特定するステップを更に有する、請求項7又は8に記載の方法。

[12] 前記配列から、あるノードに付与された整数と同じ値が格納されている連続領域を抽出することにより、前記あるノードの全ての子ノードを特定するステップを更に有する、請求項9又は10に記載の方法。

[13] ツリー型データ構造を記憶装置上に構築する方法であって、

ルート・ノードから始めて全てのノードに連続的に変化する整数を一意に割り当てるステップと、

ノード間に親子関係を定義するステップと、

を含み、

前記全てのノードに整数を一意に割り当てるステップは、

同じ世代のノードよりも子ノードを優先して番号を付与する深さ優先モードと、子ノードよりも同じ世代のノードを優先して番号を付与する幅優先モードのどちらのモードでノードに番号を付与するかを選択するステップと、

前記深さ優先モードが選択された場合に、深さ優先でノードを検索し、検索された順にノードに番号を付与するステップと、

前記幅優先モードが選択された場合に、幅優先でノードを検索し、検索された順にノードに番号を付与するステップと、

を含み、

前記ノード間に親子関係を定義するステップは、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納するステップを含む、

方法。

[14] 前記ノード間に親子関係を定義するステップは、

子ノードから親ノードへの関係を定義する子親表現モードと、親ノードから子ノードへの関係を定義する親子表現モードのどちらのモードで親子関係を定義するかを選択するステップと、

前記子親表現モードが選択された場合に、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納するステップと、

前記親子表現モードが選択された場合に、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号の配列を前記記憶装置に格納するステップと、

を含む、請求項13に記載の方法。

[15] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法であって、

前記親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより定義され、

同じ世代のノードよりも子ノードを優先してノードに番号を付与する深さ優先モードで表現されたツリー型データ構造の各ノードの世代を判定し、世代毎に属しているノードの個数を計数するステップと、

前記世代毎に属しているノードの個数に基づいて、子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで番号を付与する際に各世代において付与される番号を決定するステップと、

前記判定されたノードの世代及び前記決定された各世代において付与される番号に基づいて、前記各ノードの番号を前記幅優先モードで付与される番号に変換する変換配列を作成するステップと、

前記各ノードの前記親子関係を、前記変換配列を使用して前記幅優先モードで付与される番号で表現された親子関係に変換するステップと、

を有する方法。

[16] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法であって、

前記親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより定義され、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで

表現されたツリー型データ構造の各ノードの子孫の数を計数するステップと、

親ノードに付与されるべき番号に、当該ノードと同一の親ノードから派生した兄弟ノードのうち当該ノードよりも前に番号が付与されている兄ノードの数及び当該兄ノードの子孫の数を加算することにより、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成するステップと、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換するステップと、

を有する方法。

[17] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法であって、

前記親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより定義され、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードを深さ優先で検索し、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成するステップと、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換するステップと、

を有する方法。

[18] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法であって、

前記親子関係は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を第1の配列の要素として前記記憶装置に格納することにより定義され、

各ノードに関して、当該ノードに付与された番号が前記第1の配列の要素として出現する回数を計数するステップと、

前記各ノードに関して当該ノードに対応する子ノードに付与された番号を格納する

ため、前記計数された回数に応じた個数の連続領域を前記記憶領域に第2の配列として確保するステップと、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する子ノードの番号を順次に格納するステップと、

を有する方法。

[19] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する方法であって、

前記親子関係は、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号を第1の配列の要素として前記記憶装置に格納することにより定義され、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を格納するため、前記記憶装置に第2の配列を確保するステップと、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する親ノードの番号を順次に格納するステップと、

を有する方法。

[20] ツリー型データ構造を記憶装置上に構築する情報処理装置であって、

ルート・ノードを含むノードに固有のノード識別子を付与するノード定義手段と、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与されたノード識別子に、前記非ルート・ノードの各々の親ノードに付与されたノード識別子を関連付ける親子関係定義手段と、

を含む情報処理装置。

[21] 前記ノード定義手段は前記ノード識別子として数値を用いる、請求項20に記載の情報処理装置。

[22] 前記ノード定義手段は前記ノード識別子として連続する整数を用いる、請求項20に記載の情報処理装置。

[23] 前記親子関係定義手段は、前記非ルート・ノードの各々に付与されたノード識別子

に関連付けられた前記親ノードに付与されたノード識別子の配列を前記記憶装置に格納する、請求項20に記載の情報処理装置。

[24] ツリー型データ構造を記憶装置上に構築する情報処理装置であって、

同じ世代のノードよりも子ノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義手段と、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義手段と、

を含む情報処理装置。

[25] 前記ノード定義手段は、

最初にルート・ノードに番号を付与する手段と、

既に番号が付与されたあるノードに唯一の子ノードが存在する場合には、当該子ノードに当該あるノードに付与された前記番号の次の番号を付与する手段と、

既に番号が付与されたあるノードに複数の子ノードが存在する場合には、当該複数の子ノードの間の兄弟関係に従って、弟ノードは直上の兄ノードの全ての子孫ノードに番号が付与された後に次の番号が付与されるように、一番上の兄ノードから一番下の弟ノードまで番号を付与する手段と、

を含む、

請求項24に記載の情報処理装置。

[26] ツリー型データ構造を記憶装置上に構築する情報処理装置であって、

子ノードよりも同じ世代のノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義手段と、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義手段と、

を含む情報処理装置。

[27] 前記ノード定義手段は、

各ノードが前記ルート・ノードから何世代目のノードであるか、及び、各世代に含ま

れるノード数を算出する手段と、

最初に前記ルート・ノードに番号を付与する手段と、

ある世代に含まれる全てのノードに番号が付与されたならば、当該ある世代の次の世代にノードが存在しなくなるまで、当該次の世代に含まれる全てのノードに対して、親ノードが異なる場合には、当該親ノードに番号が付与された順番に当該ノードに番号を付与し、当該親ノードが同一である場合には、当該親ノードから派生した複数の子ノードの間に兄弟関係を定義し、一番上の兄ノードから一番下の弟ノードまで直前に付与された番号の次の番号から連続的に変化する固有の整数を順に付与する手段と、

を含む、

請求項26に記載の情報処理装置。

[28] 前記配列から、あるノードに付与された整数以上の値が格納されている連続領域を抽出することにより、前記あるノードの全ての子孫ノードを特定する手段を更に有する、請求項24又は25に記載の情報処理装置。

[29] 前記配列から、あるノードに付与された整数と同じ値が格納されている連続領域を抽出することにより、前記あるノードの全ての子ノードを特定する手段を更に有する、請求項26又は27に記載の情報処理装置。

[30] ツリー型データ構造を記憶装置上に構築する情報処理装置であって、

ルート・ノードから始めて全てのノードに連続的に変化する整数を一意に割り当てる手段と、

ノード間に親子関係を定義する手段と、

を含み、

前記全てのノードに整数を一意に割り当てる手段は、

同じ世代のノードよりも子ノードを優先して番号を付与する深さ優先モードと、子ノードよりも同じ世代のノードを優先して番号を付与する幅優先モードのどちらのモードでノードに番号を付与するかを選択する手段と、

前記深さ優先モードが選択された場合に、深さ優先でノードを検索し、検索された順にノードに番号を付与する手段と、

前記幅優先モードが選択された場合に、幅優先でノードを検索し、検索された順にノードに番号を付与する手段と、

を含み、

前記ノード間に親子関係を定義する手段は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納する手段を含む、

情報処理装置。

[31] 前記ノード間に親子関係を定義する手段は、

子ノードから親ノードへの関係を定義する子親表現モードと、親ノードから子ノードへの関係を定義する親子表現モードのどちらのモードで親子関係を定義するかを選択する手段と、

前記子親表現モードが選択された場合に、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納する手段と、

前記親子表現モードが選択された場合に、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号の配列を前記記憶装置に格納する手段と、

を含む、

請求項30に記載の情報処理装置。

[32] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する情報処理装置であって、

前記記憶装置は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに番号を付与することにより定義された前記親子関係を保持し、

同じ世代のノードよりも子ノードを優先してノードに番号を付与する深さ優先モードで表現されたツリー型データ構造の各ノードの世代を判定し、世代毎に属しているノードの個数を計数する手段と、

前記世代毎に属しているノードの個数に基づいて、子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで番号を付与する際に各世代にお

いて付与される番号を決定する手段と、

前記判定されたノードの世代及び前記決定された各世代において付与される番号に基づいて、前記各ノードの番号を前記幅優先モードで付与される番号に変換する変換配列を作成する手段と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記幅優先モードで付与される番号で表現された親子関係に変換する手段と、

を有する情報処理装置。

[33] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する情報処理装置であって、

前記記憶装置は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに番号を付与することにより定義された前記親子関係を保持し、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードの子孫の数を計数する手段と、

親ノードに付与されるべき番号に、当該ノードと同一の親ノードから派生した兄弟ノードのうち当該ノードよりも前に番号が付与されている兄ノードの数及び当該兄ノードの子孫の数を加算することにより、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成する手段と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換する手段と、

を有する情報処理装置。

[34] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する情報処理装置であって、

前記記憶装置は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに番号を付与することにより定義された前記親子関係を保持し、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードを深さ優先で検索し、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与す

る深さ優先モードで付与される番号に変換する変換配列を作成する手段と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換する手段と、

を有する情報処理装置。

[35] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する情報処理装置であって、

前記記憶装置は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を、前記親子関係を定義する第1の配列の要素として保持し、

各ノードに関して、当該ノードに付与された番号が前記第1の配列の要素として出現する回数を計数する手段と、

前記各ノードに関して当該ノードに対応する子ノードに付与された番号を格納するため、前記計数された回数に応じた個数の連続領域を前記記憶領域に第2の配列として確保する手段と、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する子ノードの番号を順次に格納する手段と、

を有する情報処理装置。

[36] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換する情報処理装置であって、

前記記憶装置は、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号を、前記親子関係を定義する第1の配列の要素として保持し、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を格納するため、前記記憶装置に第2の配列を確保する手段と、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する親ノードの番号を順次に格納する手段と、

を有する情報処理装置。

[37] ツリー型データ構造を記憶装置上に構築するコンピュータに、

ルート・ノードを含むノードに固有のノード識別子を付与するノード定義機能と、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与されたノード識別子に、前記非ルート・ノードの各々の親ノードに付与されたノード識別子を関連付ける親子関係定義機能と、

を実現させるためのプログラム。

[38] 前記ノード定義機能は前記ノード識別子として数値を用いる、請求項37に記載のプログラム。

[39] 前記ノード定義機能は前記ノード識別子として連続する整数を用いる、請求項37に記載のプログラム。

[40] 前記親子関係定義機能は、前記非ルート・ノードの各々に付与されたノード識別子に関連付けられた前記親ノードに付与されたノード識別子の配列を前記記憶装置に格納する、請求項37に記載のプログラム。

[41] ツリー型データ構造を記憶装置上に構築するコンピュータに、

同じ世代のノードよりも子ノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義機能と、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義機能と、

を実現させるためのプログラム。

[42] 前記ノード定義機能は、

最初にルート・ノードに番号を付与する機能と、

既に番号が付与されたあるノードに唯一の子ノードが存在する場合には、当該子ノードに当該あるノードに付与された前記番号の次の番号を付与する機能と、

既に番号が付与されたあるノードに複数の子ノードが存在する場合には、当該複数の子ノードの間の兄弟関係に従って、弟ノードは直上の兄ノードの全ての子孫ノードに番号が付与された後に次の番号が付与されるように、一番上の兄ノードから一番下の弟ノードまで番号を付与する機能と、

を含む、請求項41に記載のプログラム。

[43] ツリー型データ構造を記憶装置上に構築するコンピュータに、

子ノードよりも同じ世代のノードを優先して、ルート・ノードを含むノードに固有の連続する整数を付与するノード定義機能と、

前記ルート・ノード以外のノードである非ルート・ノードの各々に付与された整数の順に、前記非ルート・ノードの各々の親ノードに付与された整数を並べることにより形成される配列を前記記憶装置に格納する親子関係定義機能と、

を実現させるためのプログラム。

[44] 前記ノード定義機能は、

各ノードが前記ルート・ノードから何世代目のノードであるか、及び、各世代に含まれるノード数を算出する機能と、

最初に前記ルート・ノードに番号を付与する機能と、

ある世代に含まれる全てのノードに番号が付与されたならば、当該ある世代の次の世代にノードが存在しなくなるまで、当該次の世代に含まれる全てのノードに対して、親ノードが異なる場合には、当該親ノードに番号が付与された順番に当該ノードに番号を付与し、当該親ノードが同一である場合には、当該親ノードから派生した複数の子ノードの間に兄弟関係を定義し、一番上の兄ノードから一番下の弟ノードまで直前に付与された番号の次の番号から連続的に変化する固有の整数を順に付与する機能と、

を含む、請求項43に記載のプログラム。

[45] 前記配列から、あるノードに付与された整数以上の値が格納されている連続領域を抽出することにより、前記あるノードの全ての子孫ノードを特定する機能を更に有する、請求項41又は42に記載のプログラム。

[46] 前記配列から、あるノードに付与された整数と同じ値が格納されている連続領域を抽出することにより、前記あるノードの全ての子ノードを特定する機能を更に有する、請求項43又は44に記載のプログラム。

[47] ツリー型データ構造を記憶装置上に構築するコンピュータに、

ルート・ノードから始めて全てのノードに連続的に変化する整数を一意に割り当てる機能と、

ノード間に親子関係を定義する機能と、

を実現させ、

前記全てのノードに整数を一意に割り当てる機能は、

同じ世代のノードよりも子ノードを優先して番号を付与する深さ優先モードと、子ノードよりも同じ世代のノードを優先して番号を付与する幅優先モードのどちらのモードでノードに番号を付与するかを選択する機能と、

前記深さ優先モードが選択された場合に、深さ優先でノードを検索し、検索された順にノードに番号を付与する機能と、

前記幅優先モードが選択された場合に、幅優先でノードを検索し、検索された順にノードに番号を付与する機能と、

を含み、

前記ノード間に親子関係を定義する機能は、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納する機能を含む、

プログラム。

[48] 前記ノード間に親子関係を定義する機能は、

子ノードから親ノードへの関係を定義する子親表現モードと、親ノードから子ノードへの関係を定義する親子表現モードのどちらのモードで親子関係を定義するかを選択する機能と、

前記子親表現モードが選択された場合に、子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納する機能と、

前記親子表現モードが選択された場合に、親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された番号の配列を前記記憶装置に格納する機能と、

を含む、請求項47に記載のプログラム。

[49] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換するコンピュータに、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより前記親子関係を定義する機能と、

同じ世代のノードよりも子ノードを優先してノードに番号を付与する深さ優先モードで表現されたツリー型データ構造の各ノードの世代を判定し、世代毎に属しているノードの個数を計数する機能と、

前記世代毎に属しているノードの個数に基づいて、子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで番号を付与する際に各世代において付与される番号を決定する機能と、

前記判定されたノードの世代及び前記決定された各世代において付与される番号に基づいて、前記各ノードの番号を前記幅優先モードで付与される番号に変換する変換配列を作成する機能と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記幅優先モードで付与される番号で表現された親子関係に変換する機能と、

を実現させるためのプログラム。

[50] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換するコンピュータに、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより前記親子関係を定義する機能と、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードの子孫の数を計数する機能と、

親ノードに付与されるべき番号に、当該ノードと同一の親ノードから派生した兄弟ノードのうち当該ノードよりも前に番号が付与されている兄ノードの数及び当該兄ノードの子孫の数を加算することにより、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成する機能と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換する機能と、

を実現させるためのプログラム。

[51] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換するコンピュータに、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を前記記憶装置に格納することにより前記親子関係を定義する機能と、

子ノードよりも同じ世代のノードを優先してノードに番号を付与する幅優先モードで表現されたツリー型データ構造の各ノードを深さ優先で検索し、前記幅優先モードで付与された番号を、同じ世代のノードよりも子ノードを優先してノードの番号を付与する深さ優先モードで付与される番号に変換する変換配列を作成する機能と、

前記各ノードの前記親子関係を、前記変換配列を使用して前記深さ優先モードで付与される番号で表現された親子関係に変換する機能と、

を実現させるためのプログラム。

[52] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換するコンピュータに、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を第1の配列の要素として前記記憶装置に格納することにより前記親子関係を定義する機能と、

各ノードに関して、当該ノードに付与された番号が前記第1の配列の要素として出現する回数を計数する機能と、

前記各ノードに関して当該ノードに対応する子ノードに付与された番号を格納するため、前記計数された回数に応じた個数の連続領域を前記記憶領域に第2の配列として確保する機能と、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する子ノードの番号を順次に格納する機能と、

を実現させるためのプログラム。

[53] 記憶装置上で親子関係を用いて表現されたツリー型データ構造の表現形式を変換するコンピュータに、

親ノードに付与された番号の順に、当該親ノードに対応する子ノードに付与された

番号を第1の配列の要素として前記記憶装置に格納することにより前記親子関係を定義する機能と、

子ノードに付与された番号の順に、当該子ノードに対応する親ノードに付与された番号を格納するため、前記記憶装置に第2の配列を確保する機能と、

前記第1の配列の要素を順次に読み出し、当該要素と値が一致する番号が付与されたノードのために確保された前記第2の配列の要素として、前記第1の配列の要素に対する親ノードの番号を順次に格納する機能と、

を実現させるためのプログラム。

[54] 請求項37乃至53のうちいずれか一項に記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。

[図1]

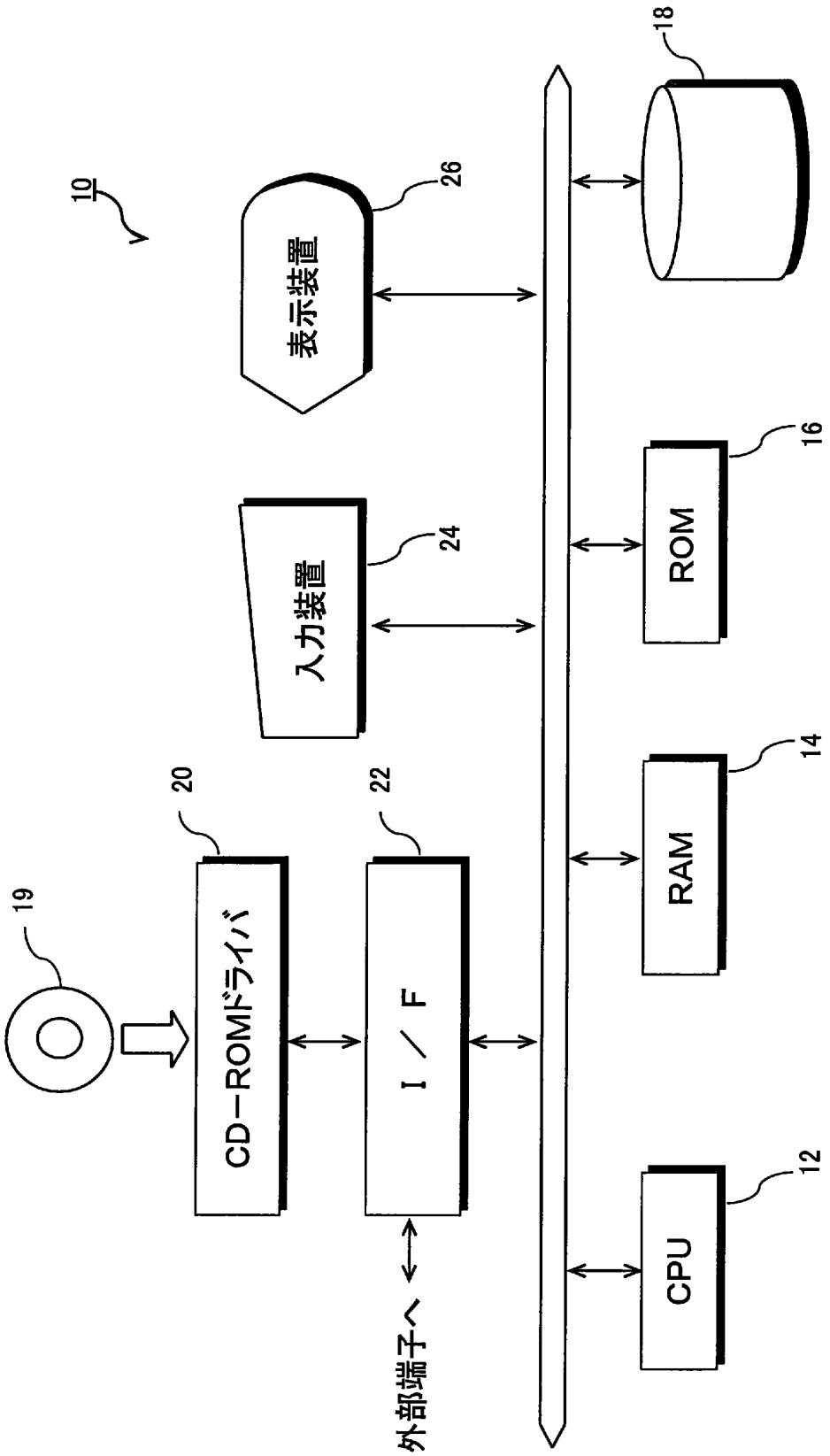

10
18
26
表示装置
入力装置
24
16
ROM
20
22
14
RAM
19
CD−ROMドライバ
I／F
外部端子へ
CPU
12

Fig.1

[図2]

Fig.2A

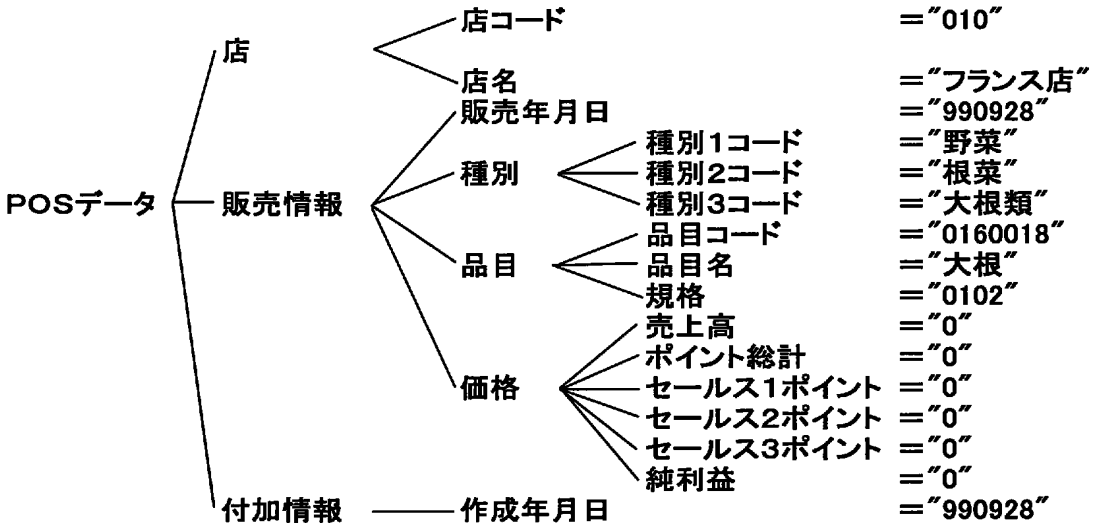


Fig.2B

```
<posdata>
   <shop>
      <shopCode>010</shopCode>
      <shopName>フランス店</shopName>
   </shop>
   <salesInformation>
      <sellDate>990928</sellDate>
      <class>
           <class1 code="01">野菜</class1>
           <class2 code="01">根菜</class2>
           <class3 code="01">大根類</class3>
      </class>
      <goods>
           <goodsCode>"0160018"</goodsCode>
           <goodsName>大根</goodsName>
           <standard>0102</standard>
      </goods>
      <price>
           <amountOfSales>0</amountOfSales>
           <amountOfPoints>0</amountOfPoints>
           <sales1 point="0">0</sales1>
           <sales2 point="0">0</sales2>
           <sales3 point="0">0</sales3>
           <grossProfit>0</grossProfit>
      </price>
   </salesInformation>
   <additionalInformation>
      <createdDate>990928</createdDate>
   </additionalInformation>
</posdata>
```

[図3]

Fig.3A

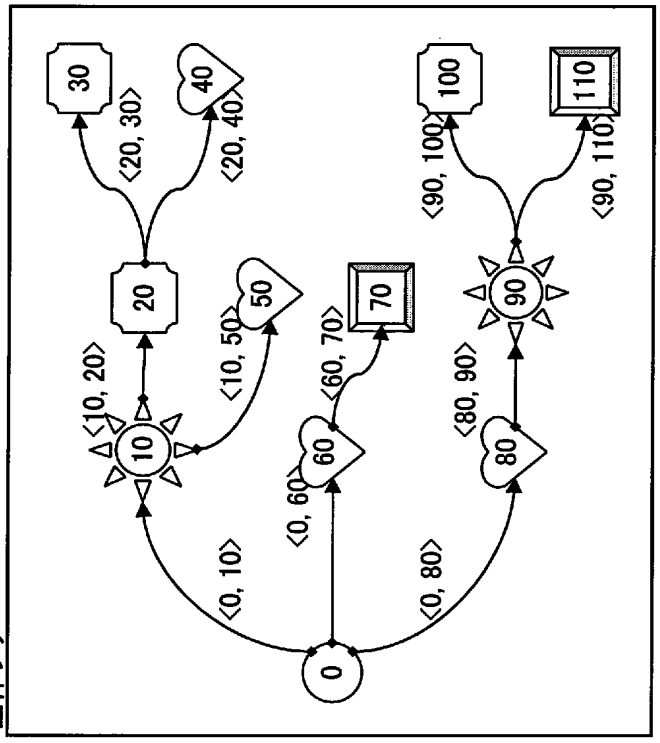


Fig.3B

アークリスト

	From-ID	To-ID
0	0	10
1	0	60
2	0	80
3	10	20
4	10	50
5	20	30
6	20	40
7	60	70
8	80	90
9	90	100
10	90	110

Fig.3C

ノードリスト

	ID	Type
0	0	Root
1	10	Sun
2	20	Sqr
3	30	Sqr
4	40	Heart
5	50	Heart
6	60	Heart
7	70	Btn
8	80	Heart
9	90	Sun
10	100	Sqr
11	110	Btn

[図4]

Fig.4A

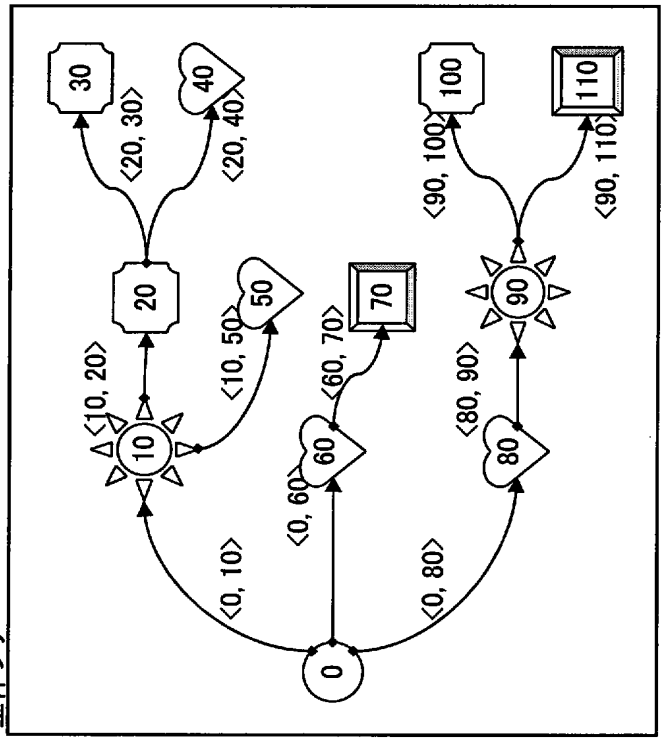


Fig.4B

アークリスト:「子→親」関係

To-ID	From-ID
0	–
10	0
20	10
30	20
40	20
50	10
60	0
70	60
80	0
90	80
100	90
110	90

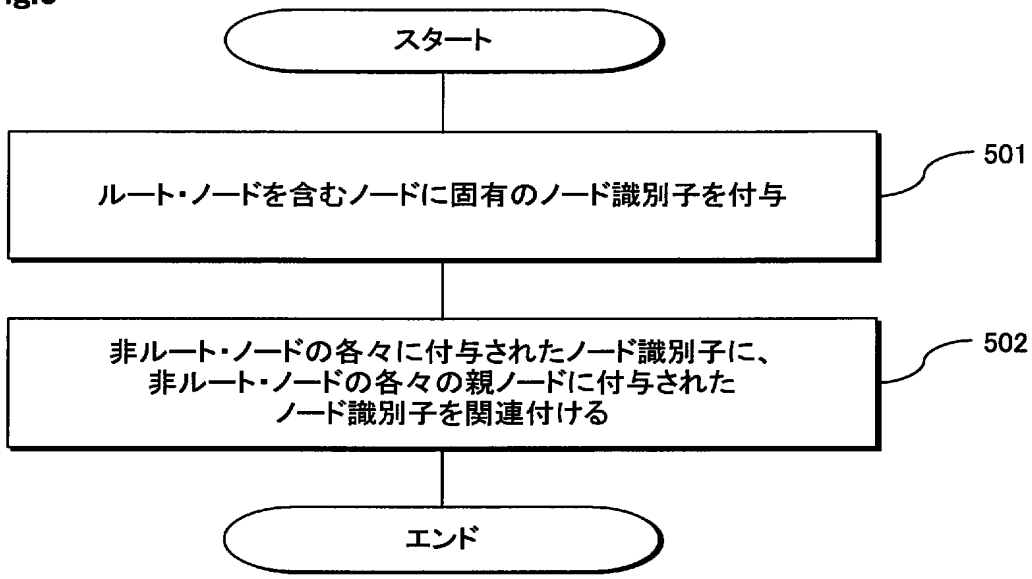
Fig.4C

アークリスト:「子→親」関係

To-ID	From-ID
10	0
20	10
30	20
40	20
50	10
60	0
70	60
80	0
90	80
100	90
110	90

[図5]

Fig.5


スタート
ルート・ノードを含むノードに固有のノード識別子を付与
501
非ルート・ノードの各々に付与されたノード識別子に、
非ルート・ノードの各々の親ノードに付与された
ノード識別子を関連付ける
502
エンド

[図6]

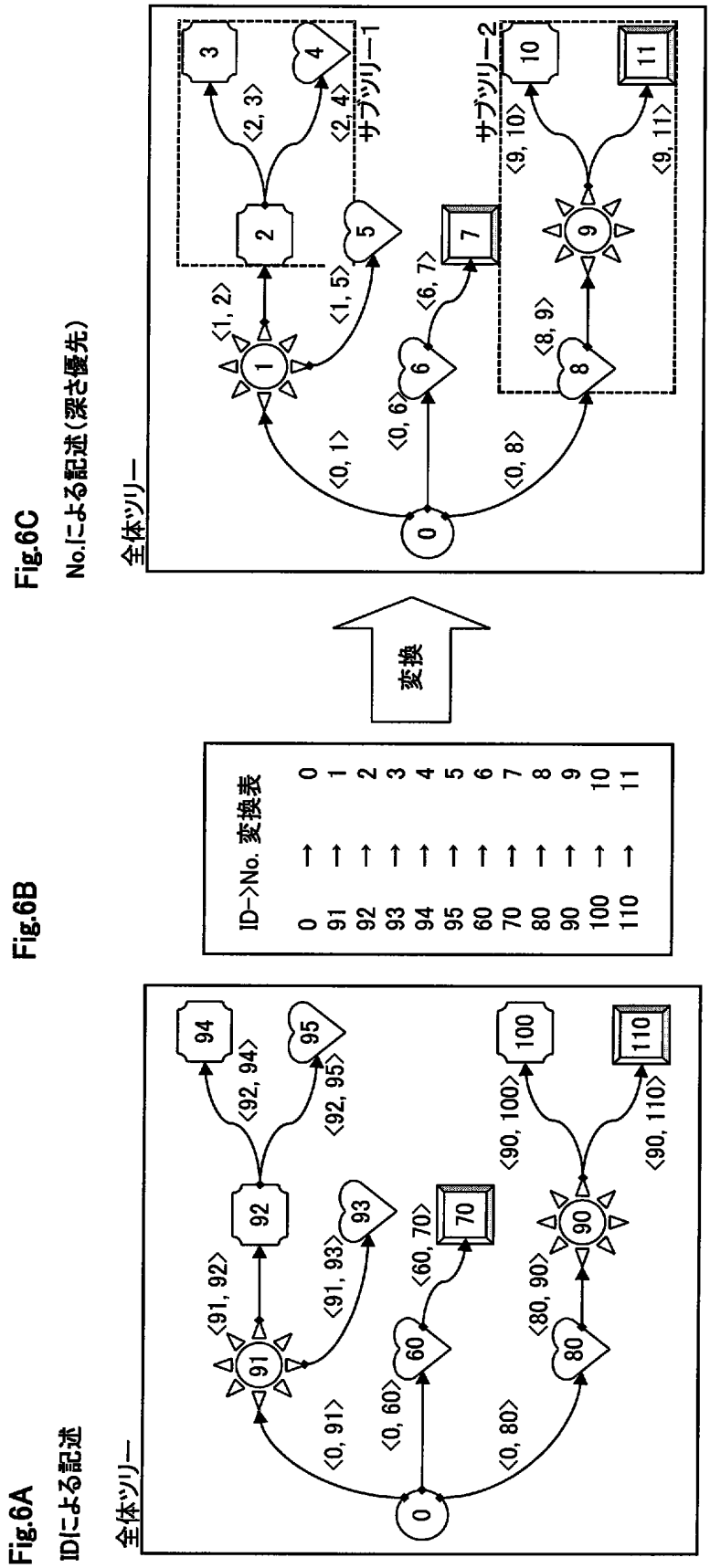


Fig.6A
IDによる記述
全体ツリー
Fig.6B
ID->No. 変換表
0 0
91 1
92 2
93 3
94 4
95 5
60 6
70 7
80 8
90 9
100 10
110 11
変換
Fig.6C
No.による記述(深さ優先)
全体ツリー
サブツリー1
サブツリー2

[図7]

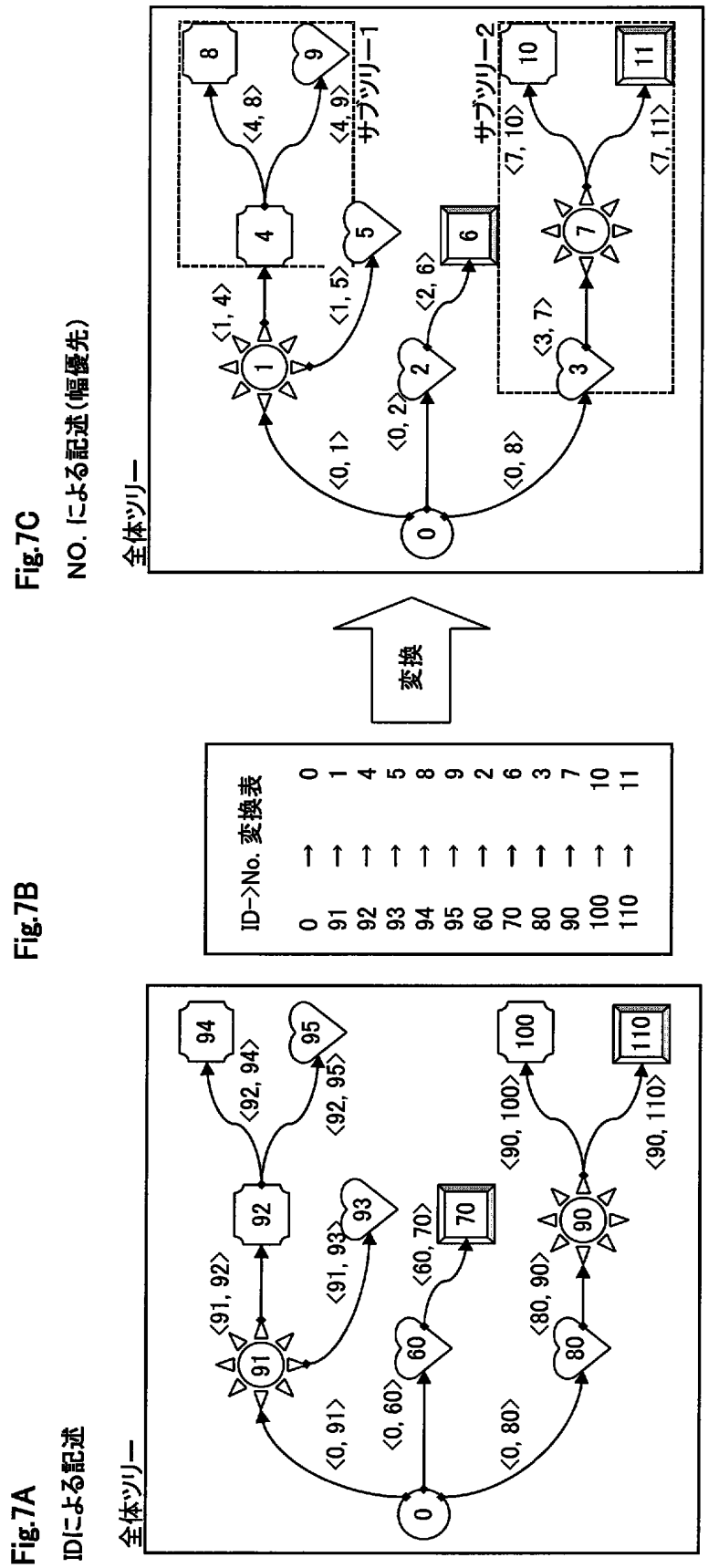
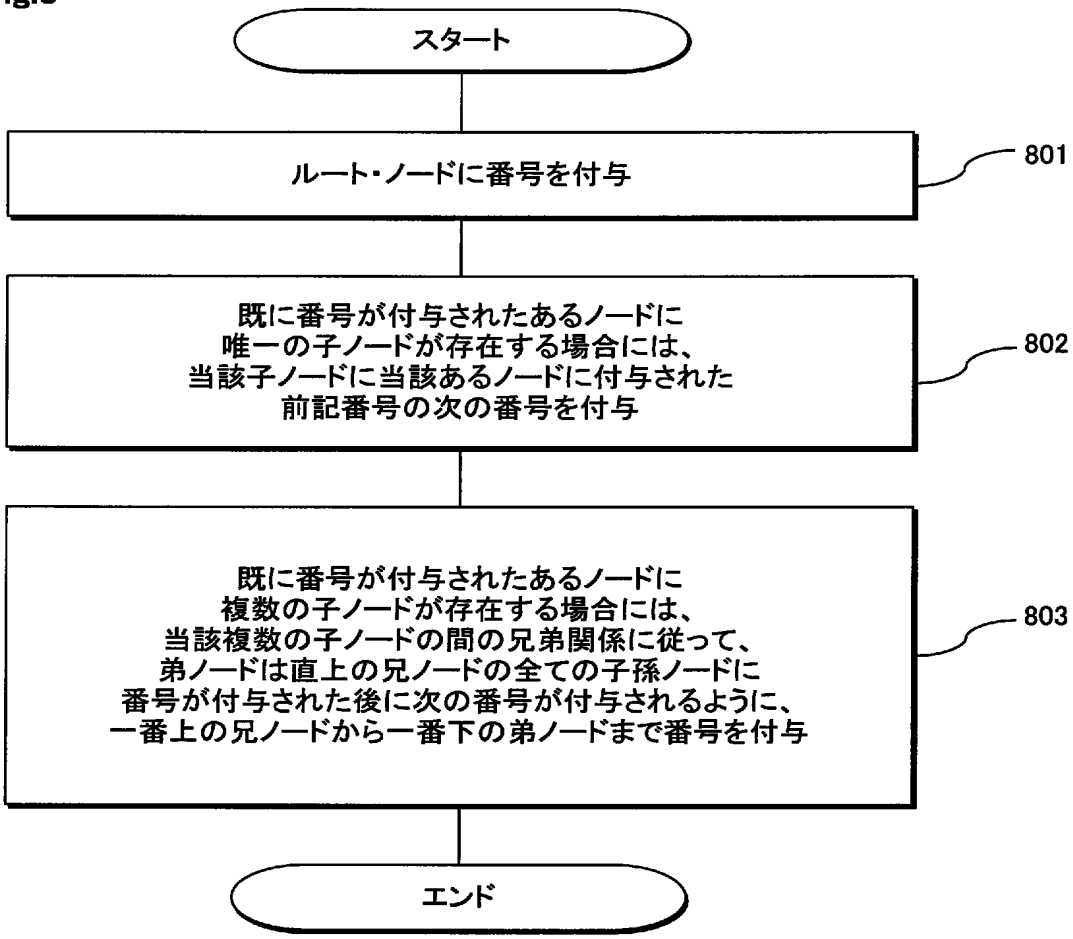


Fig.7A
IDによる記述
全体ツリー
0
91
92
93
94
95
60
70
80
90
100
110
<0, 91>
<0, 60>
<0, 80>
<91, 92>
<91, 93>
<92, 94>
<92, 95>
<60, 70>
<80, 90>
<90, 100>
<90, 110>
Fig.7B
ID->No. 変換表
0 0
91 1
92 4
93 5
94 8
95 9
60 2
70 6
80 3
90 7
100 10
110 11
変換
Fig.7C
NO. による記述(幅優先)
全体ツリー
0
1
2
3
4
5
6
7
8
9
10
11
<0, 1>
<0, 2>
<0, 8>
<1, 4>
<1, 5>
<2, 6>
<3, 7>
<4, 8>
<4, 9>
<7, 10>
<7, 11>
サブツリー1
サブツリー2

[図8]

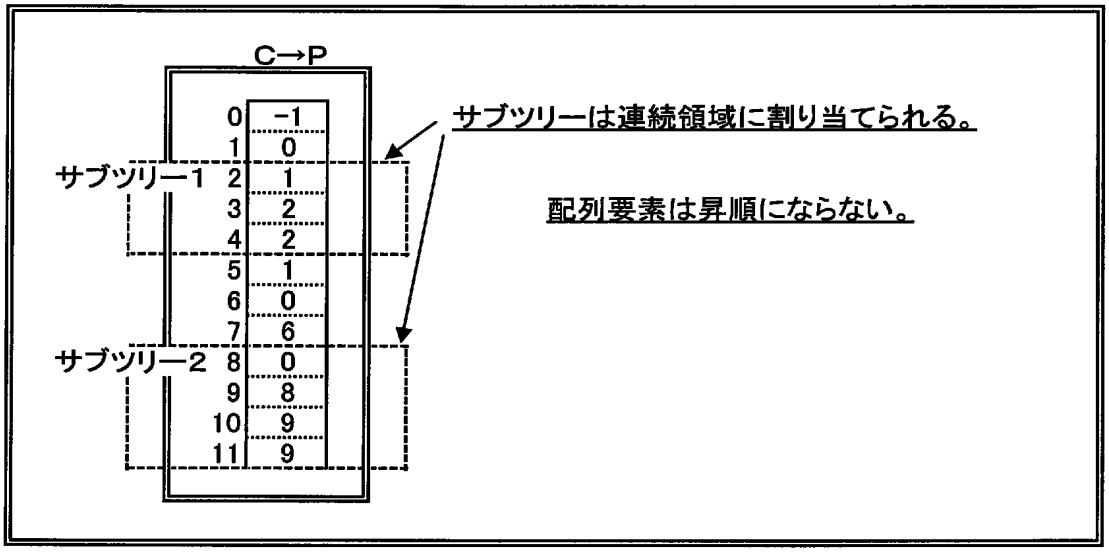
Fig.8



[図9]

Fig.9

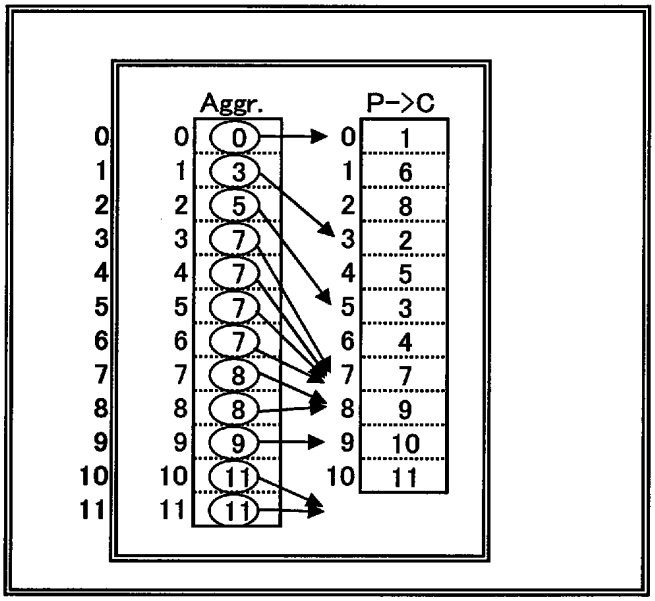
深さ優先「子→親」関係に基づく親子関係の配列



[図10]

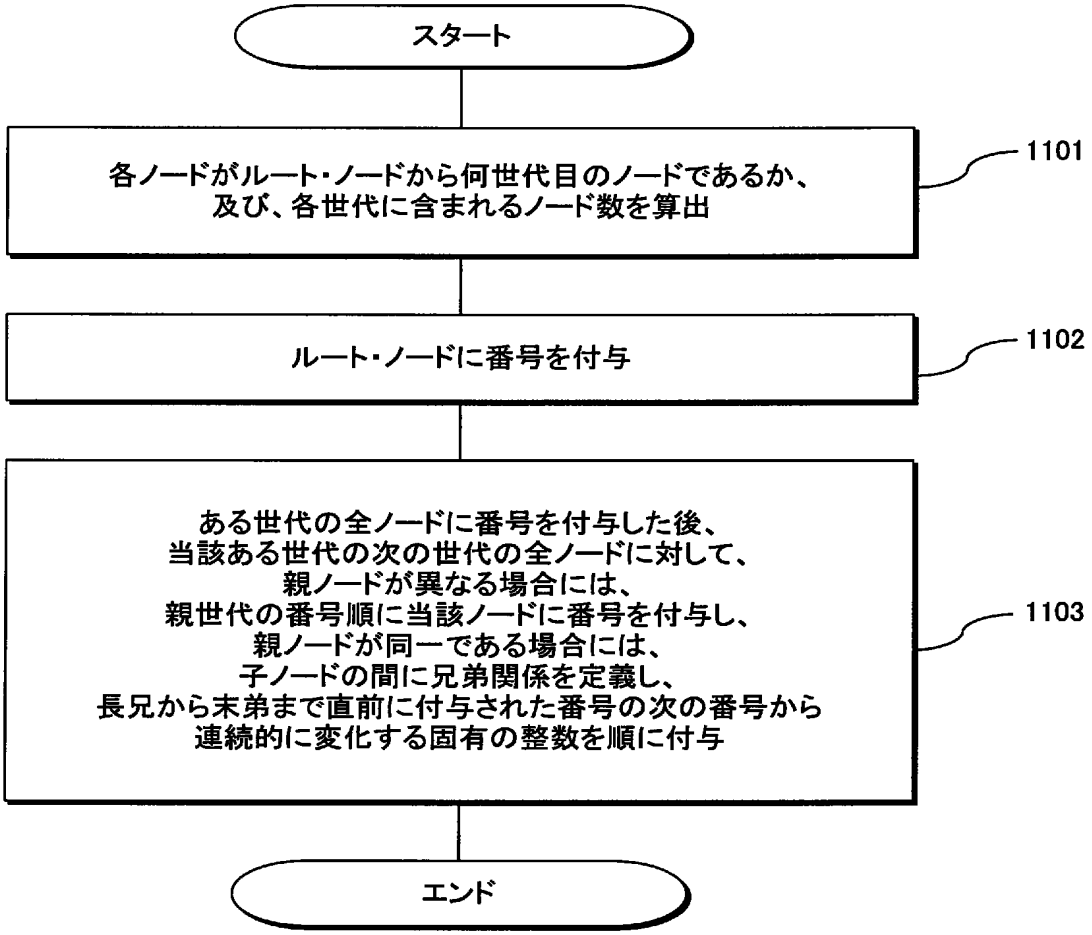
Fig.10

深さ優先「親→子」関係に基づく親子関係の配列



[図11]

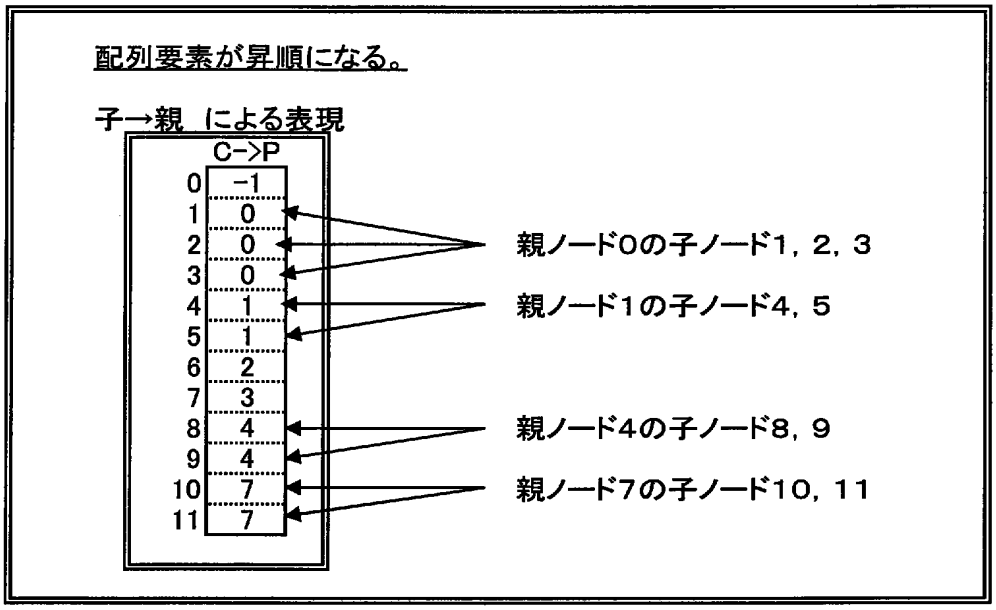
Fig.11



[図12]

Fig.12

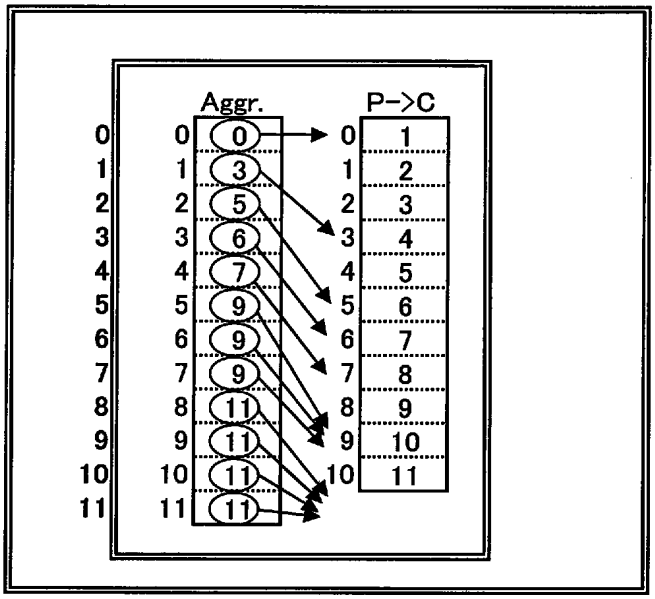
幅優先「子→親」関係に基づく親子関係の配列



[図13]

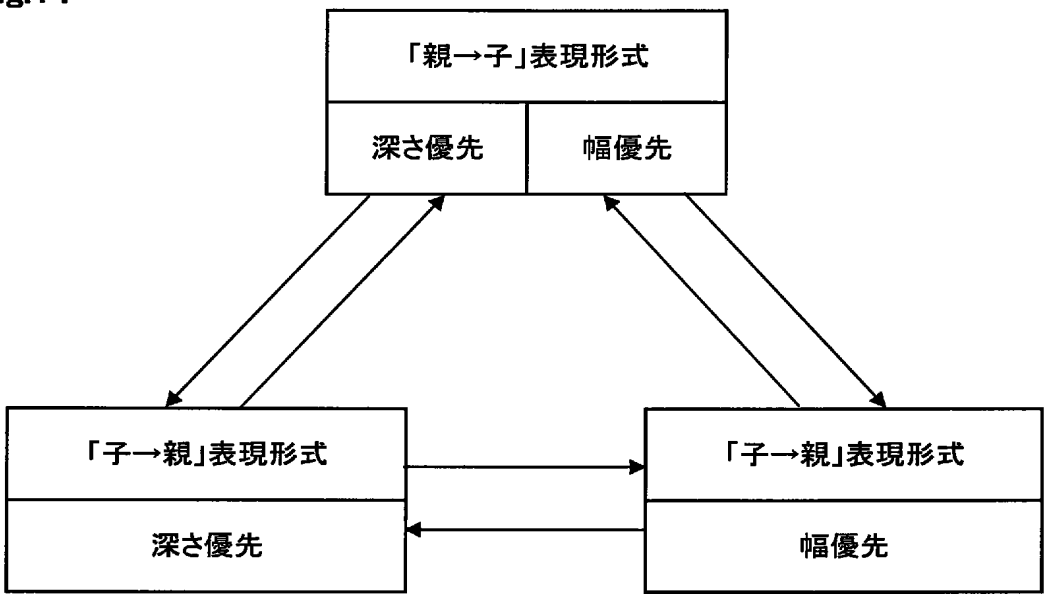
Fig.13

幅優先「親→子」関係に基づく親子関係の配列



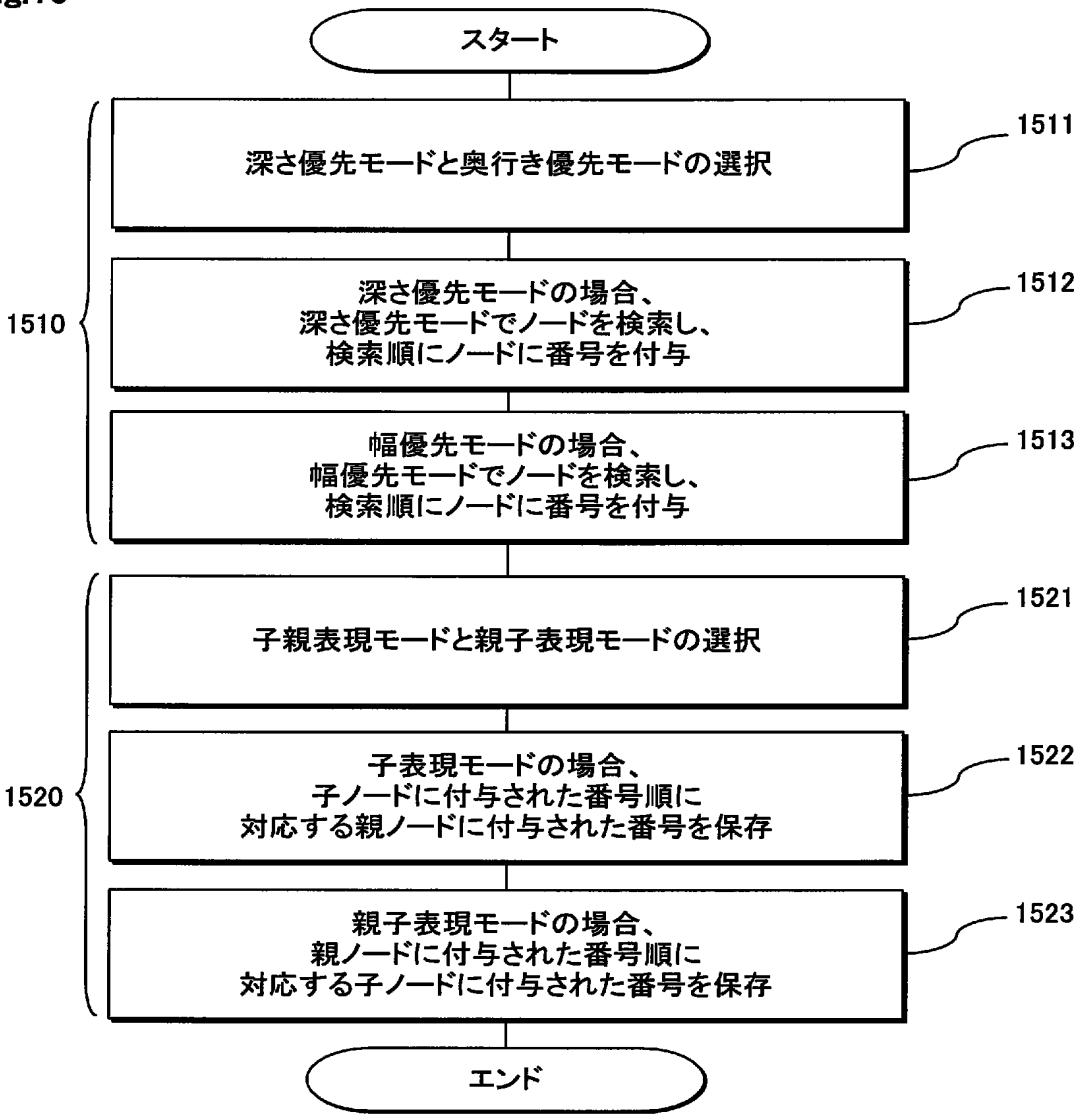
[図14]

Fig.14



[図15]

Fig.15


スタート
深さ優先モードと奥行き優先モードの選択
1511
深さ優先モードの場合、
深さ優先モードでノードを検索し、
検索順にノードに番号を付与
1512
1510
幅優先モードの場合、
幅優先モードでノードを検索し、
検索順にノードに番号を付与
1513
子親表現モードと親子表現モードの選択
1521
子表現モードの場合、
子ノードに付与された番号順に
対応する親ノードに付与された番号を保存
1522
1520
親子表現モードの場合、
親ノードに付与された番号順に
対応する子ノードに付与された番号を保存
1523
エンド

[図16]

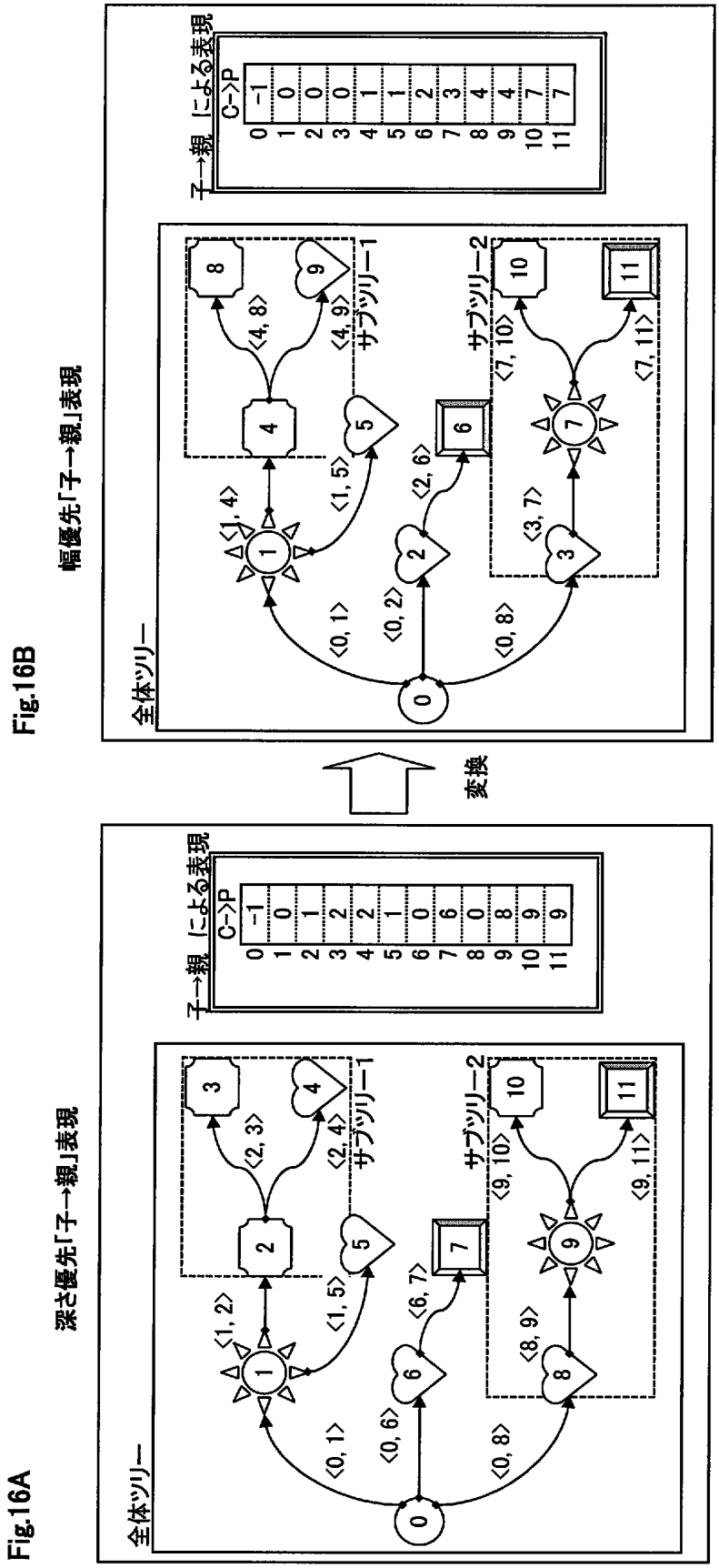
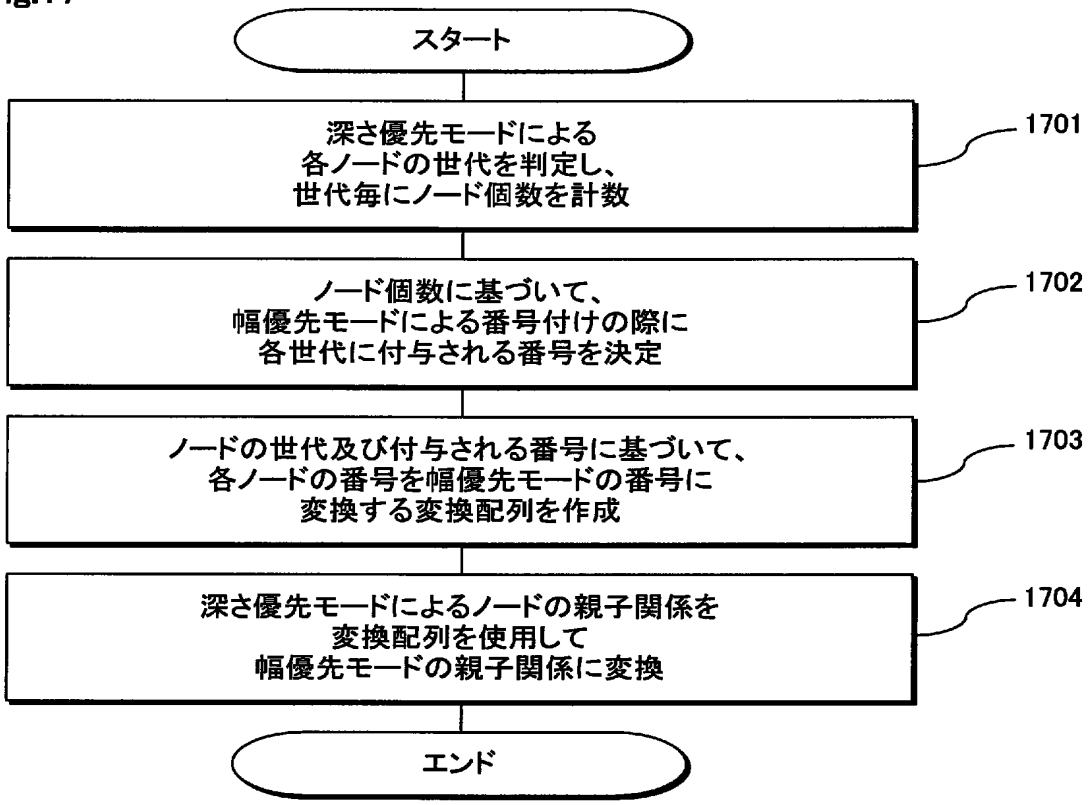


Fig.16A
深さ優先「子→親」表現
全体ツリー
子→親 による表現
C->P
変換
Fig.16B
幅優先「子→親」表現
全体ツリー
サブツリー1
サブツリー2
子→親 による表現
C->P

[図17]

Fig.17


スタート
深さ優先モードによる
各ノードの世代を判定し、
世代毎にノード個数を計数
1701
ノード個数に基づいて、
幅優先モードによる番号付けの際に
各世代に付与される番号を決定
1702
ノードの世代及び付与される番号に基づいて、
各ノードの番号を幅優先モードの番号に
変換する変換配列を作成
1703
深さ優先モードによるノードの親子関係を
変換配列を使用して
幅優先モードの親子関係に変換
1704
エンド

[図18]

Fig.18A

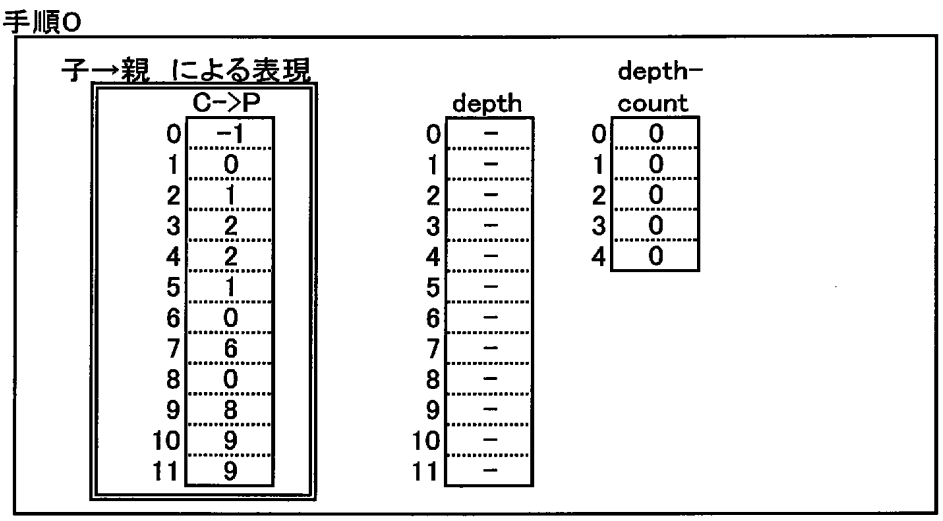


Fig.18B

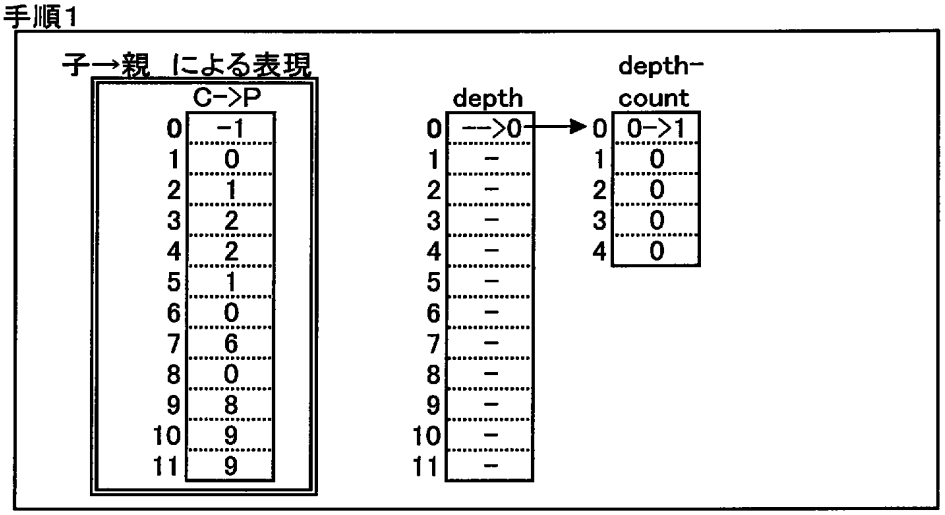
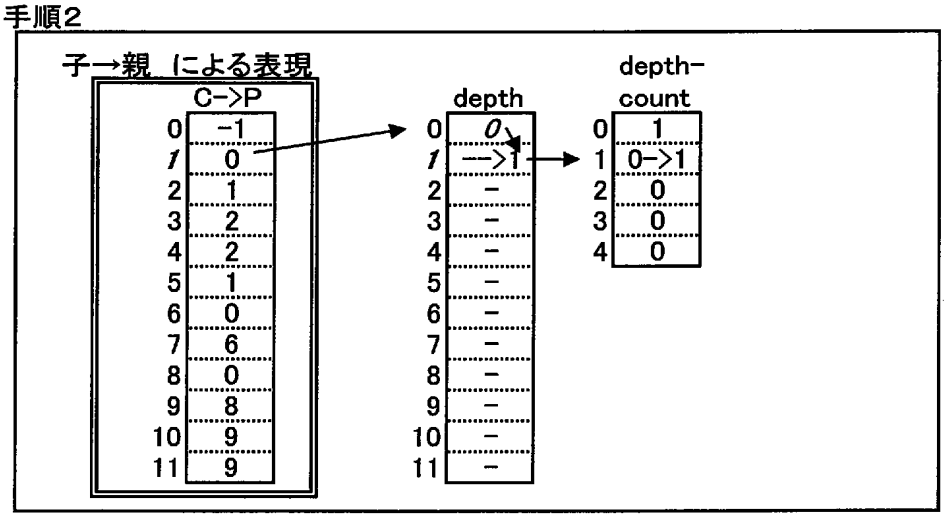


Fig.18C



[図19]

Fig.19A 手順3

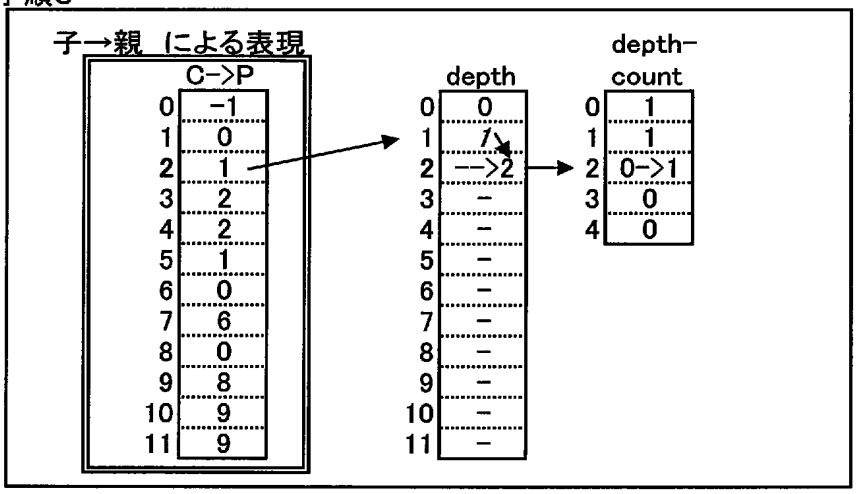


Fig.19B 手順4

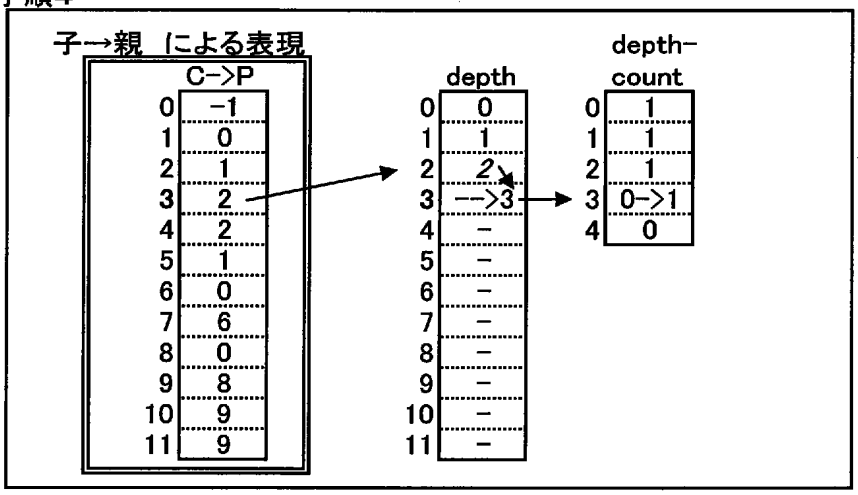
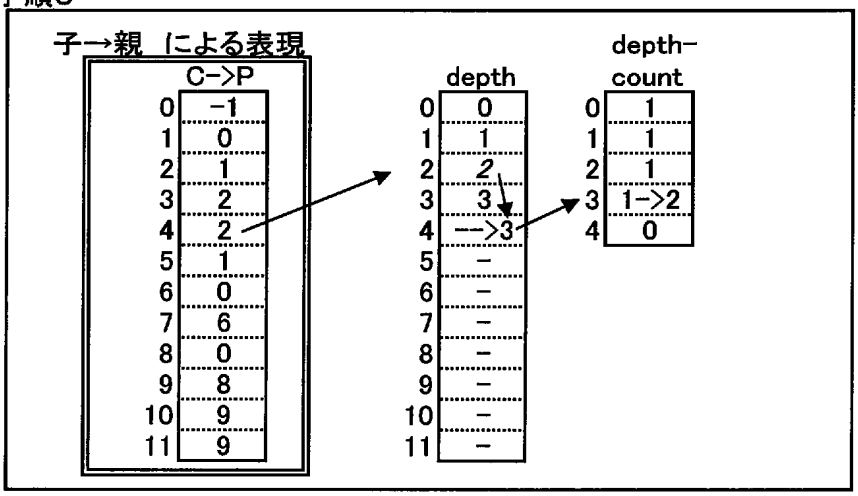


Fig.19C 手順5



[図20]

Fig.20A

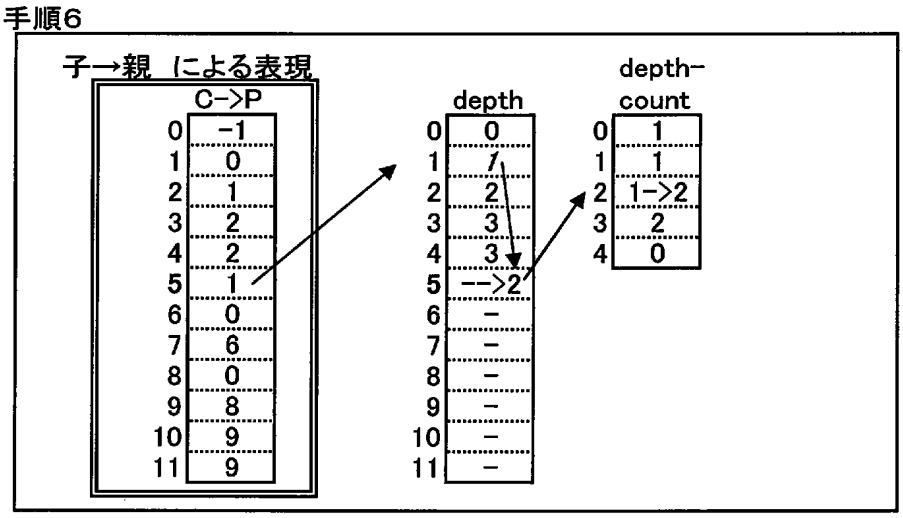


Fig.20B

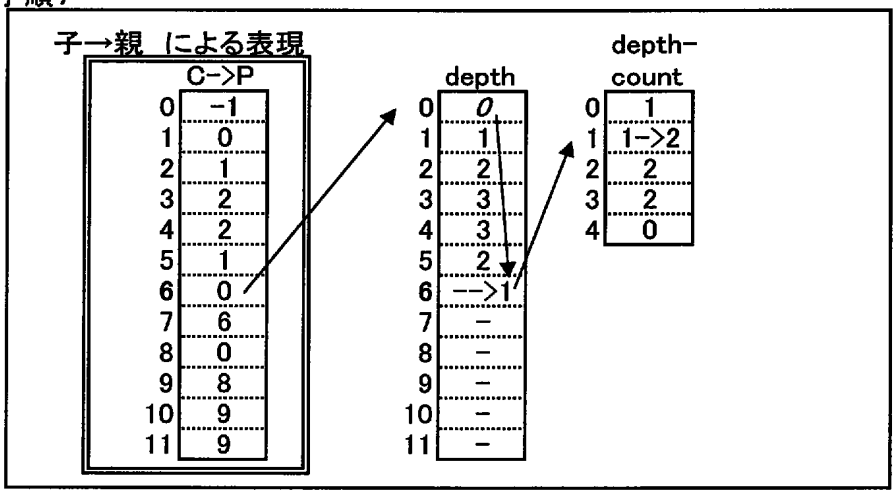
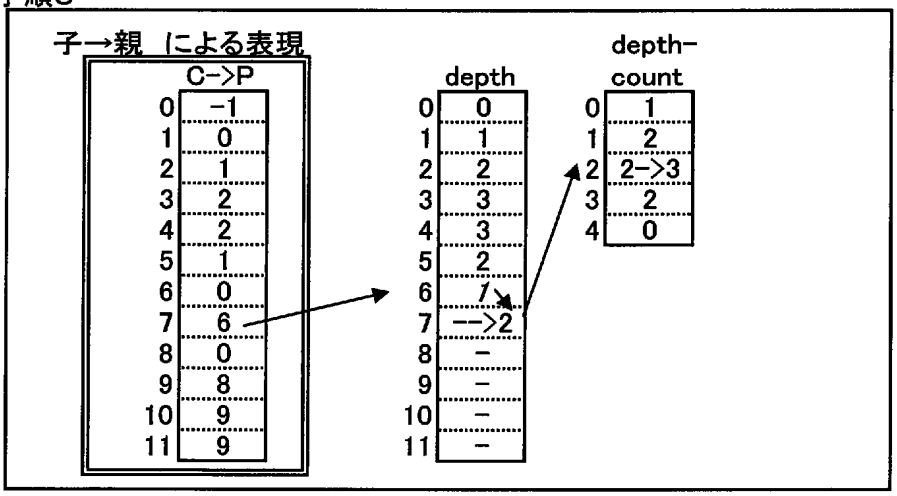


Fig.20C



[図21]

Fig.21A 手順9

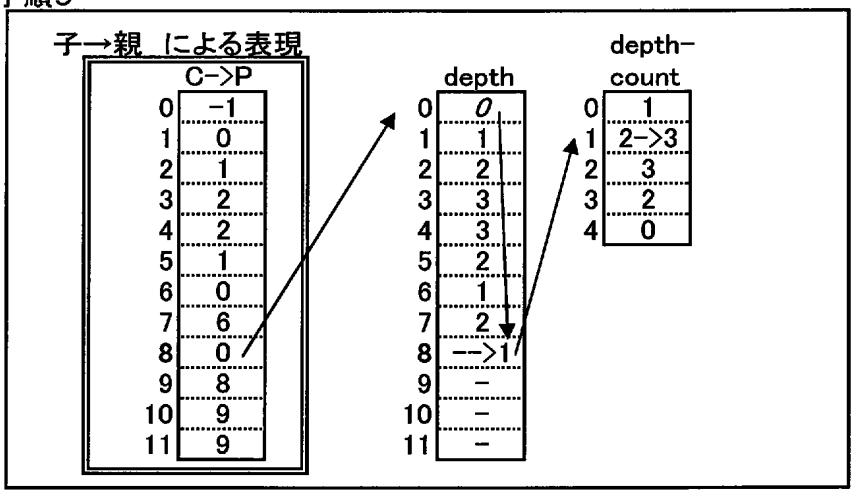


Fig.21B 手順10

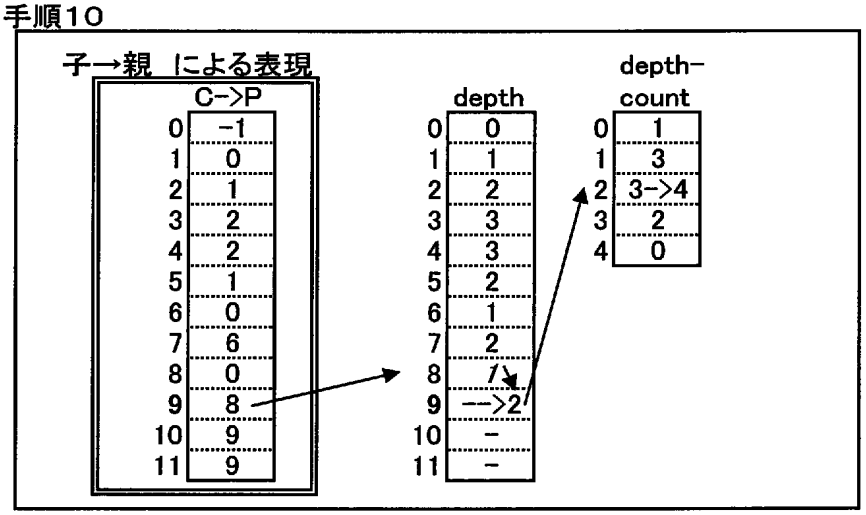
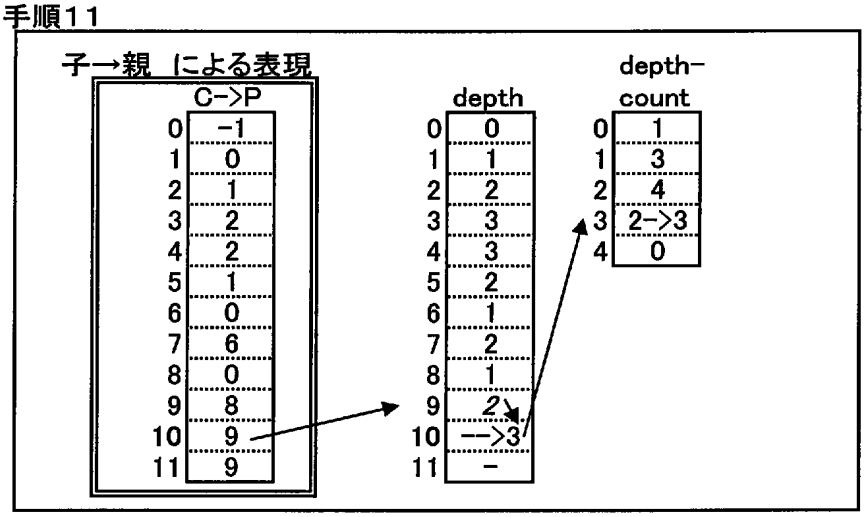
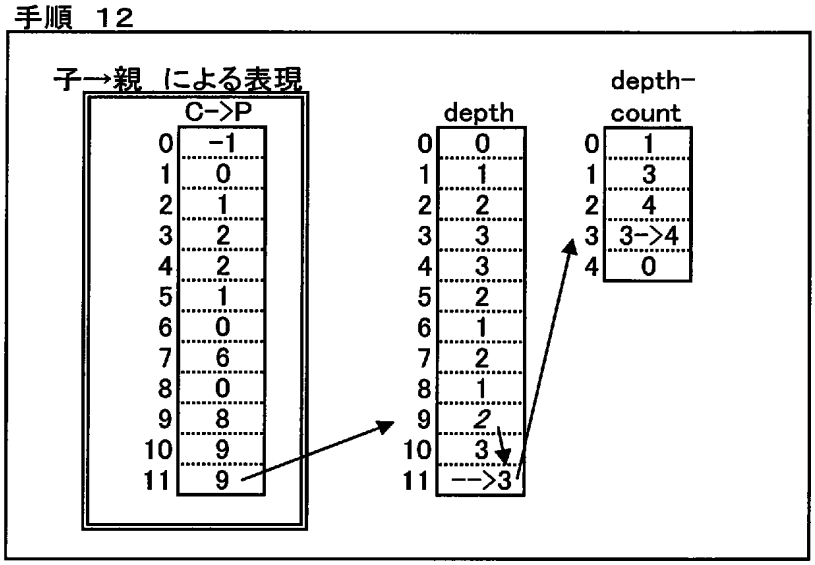


Fig.21C 手順11



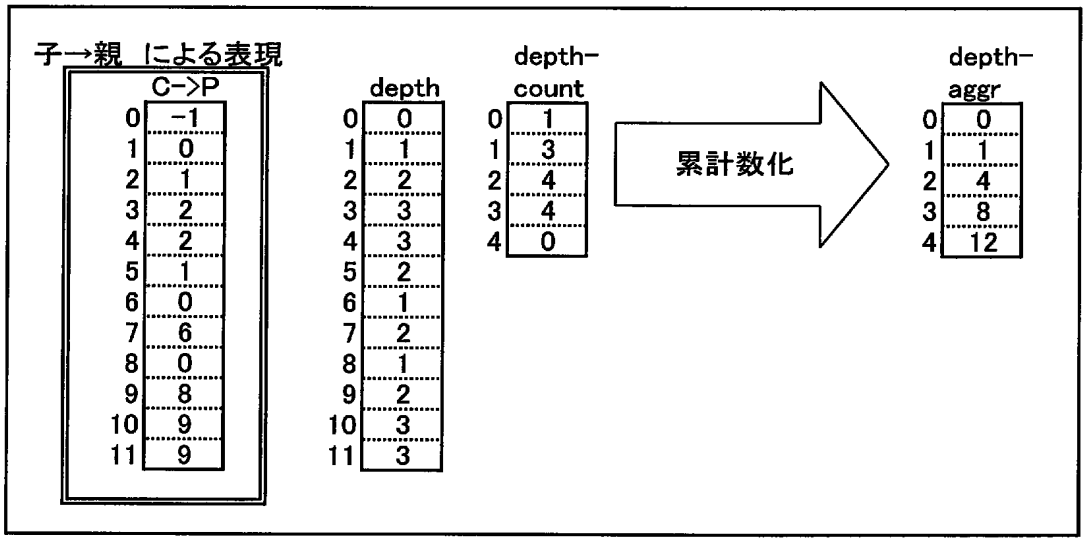
[図22]

Fig.22



[図23]

Fig.23



[図24]

Fig.24A

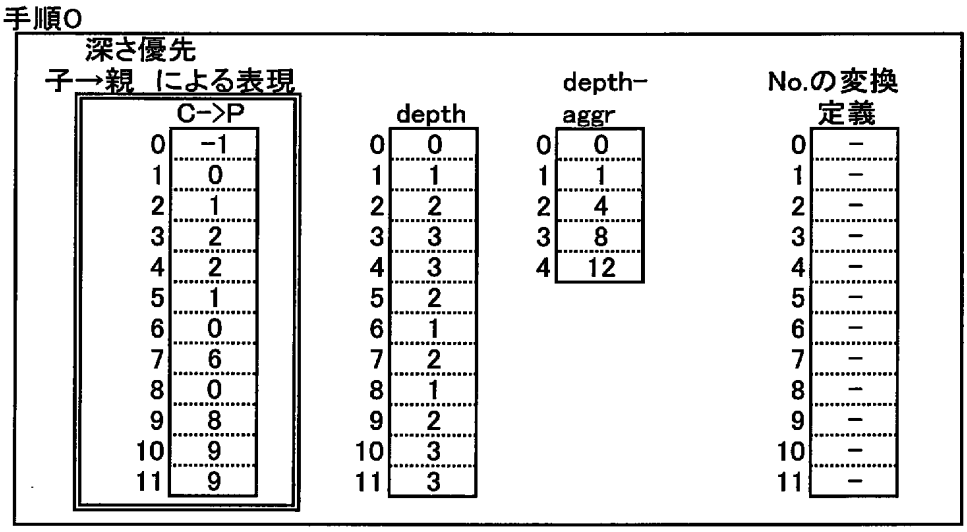


Fig.24B

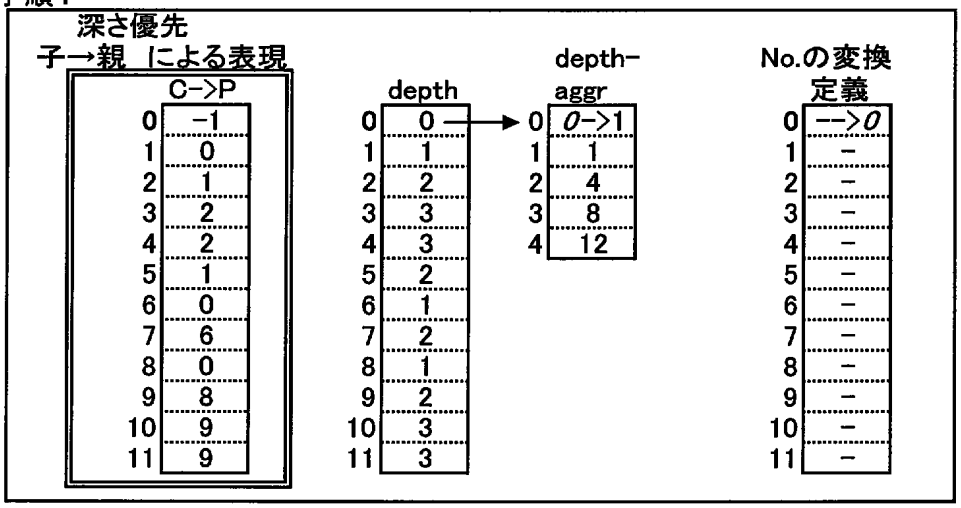
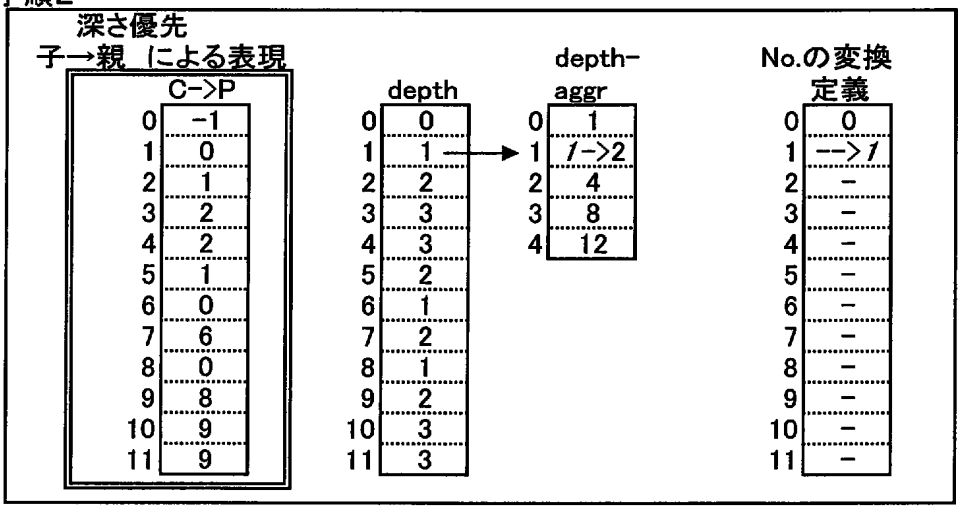


Fig.24C



[図25]

Fig.25A

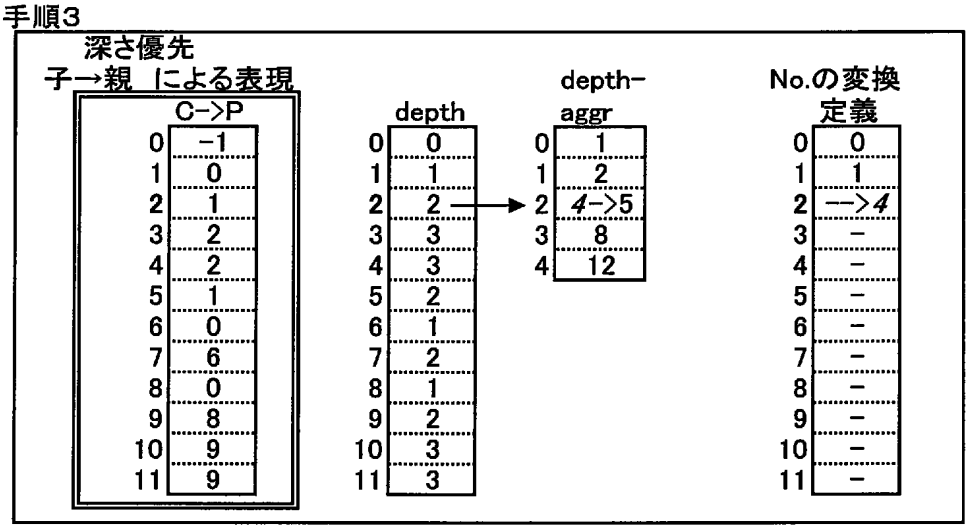


Fig.25B

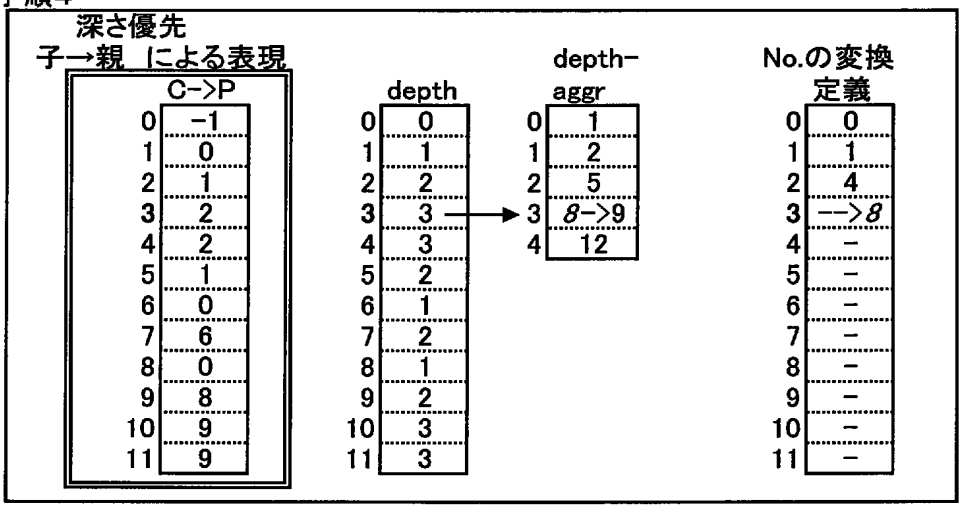
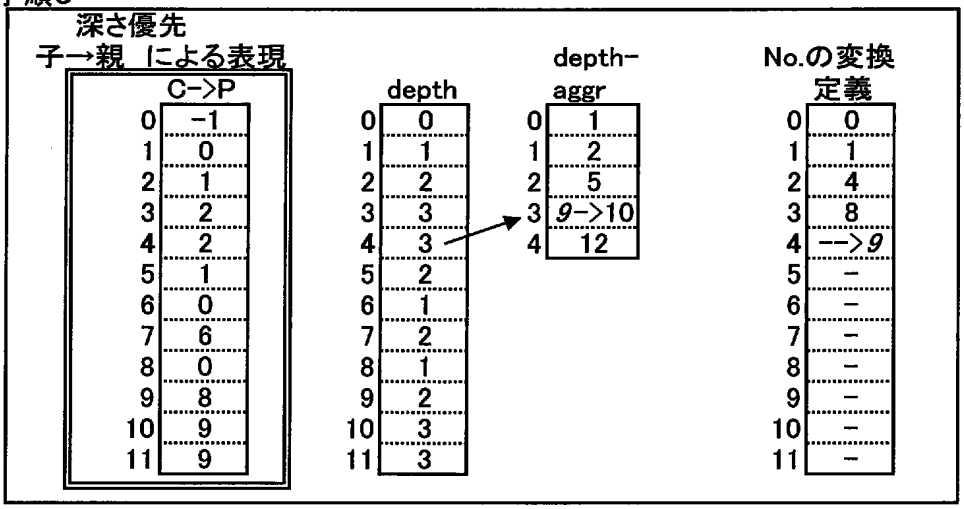


Fig.25C



[図26]

Fig.26A

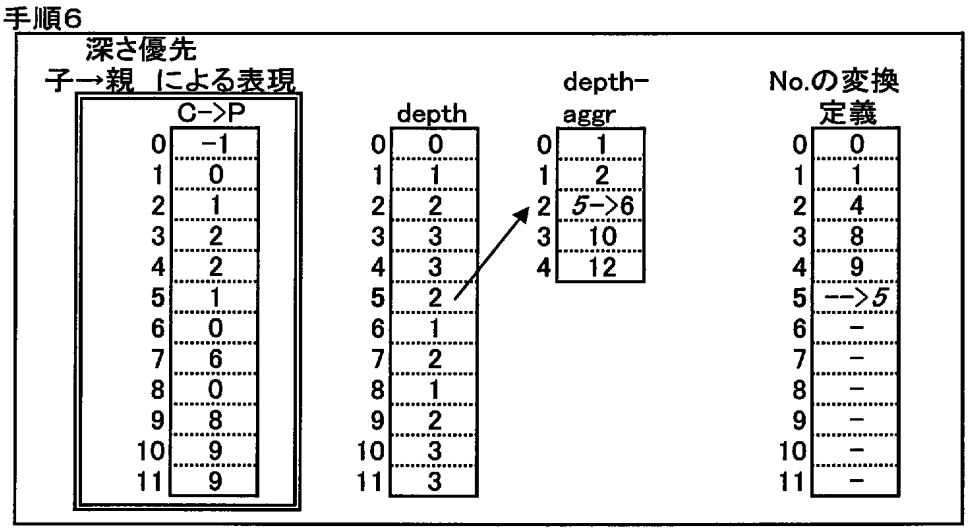


Fig.26B

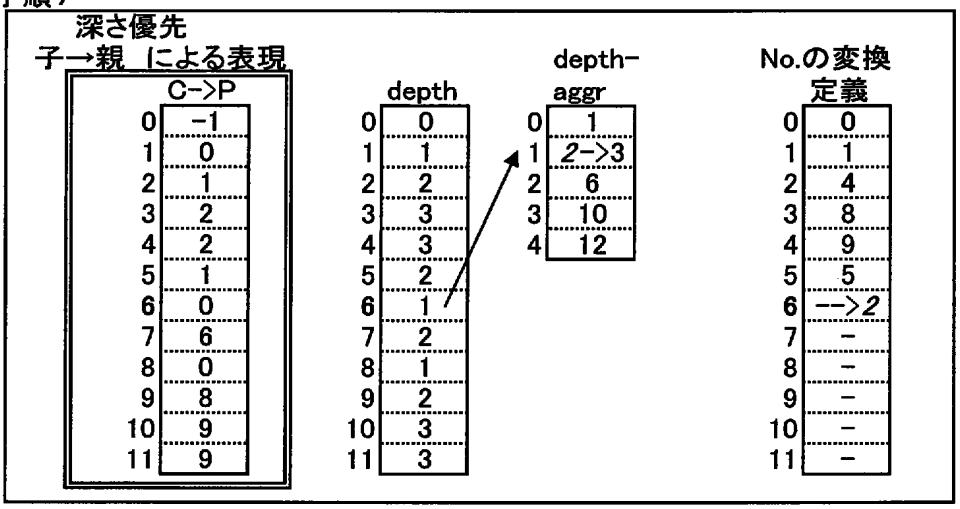
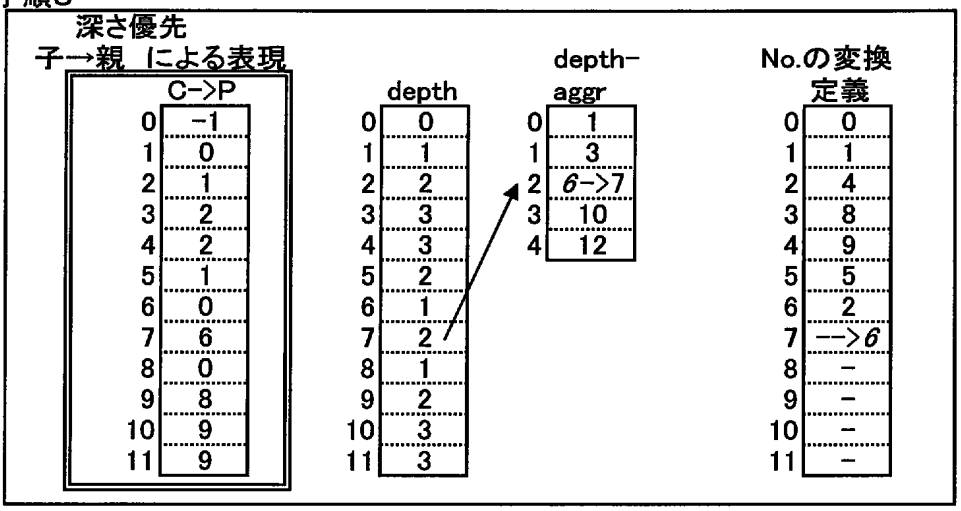


Fig.26C



[図27]

Fig.27A

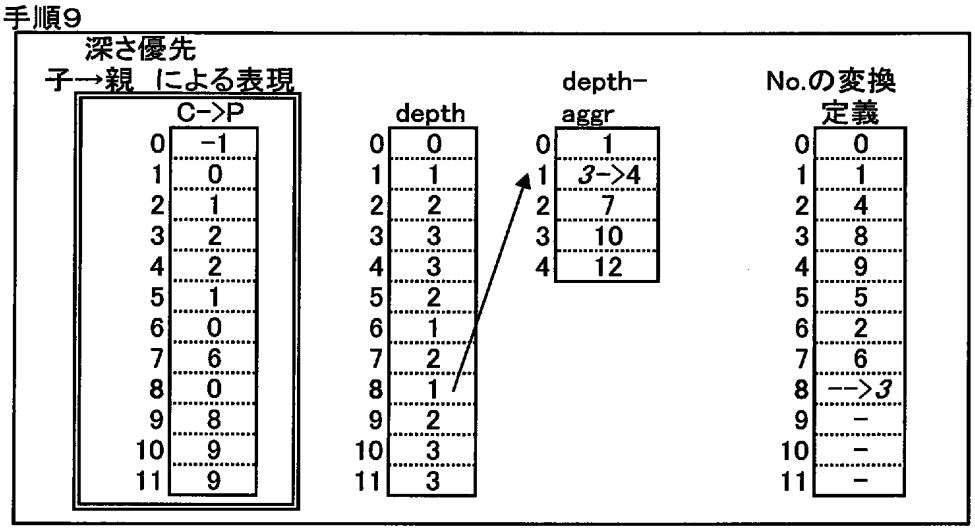


Fig.27B

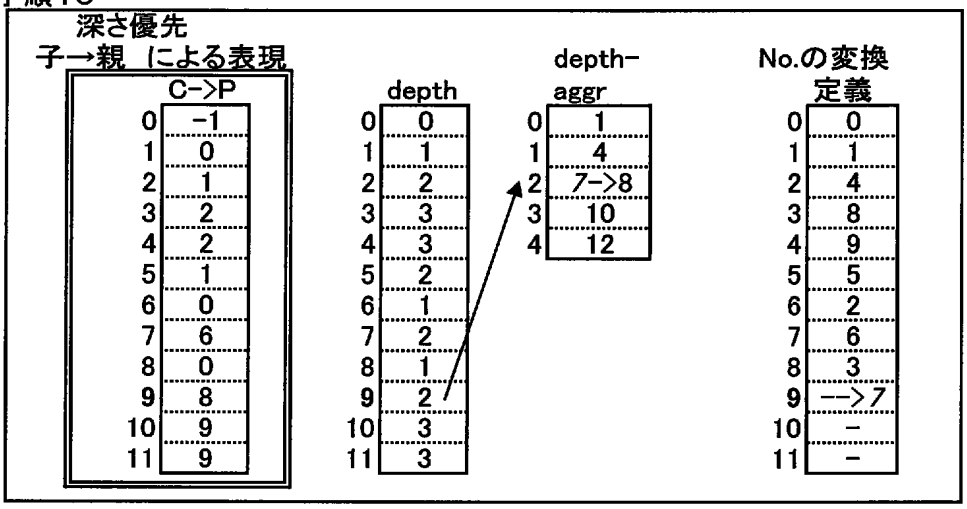
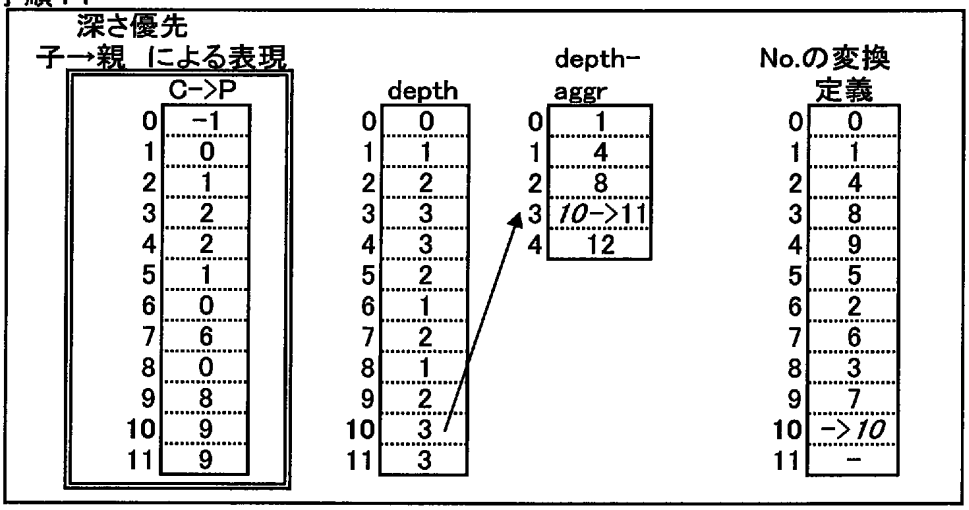
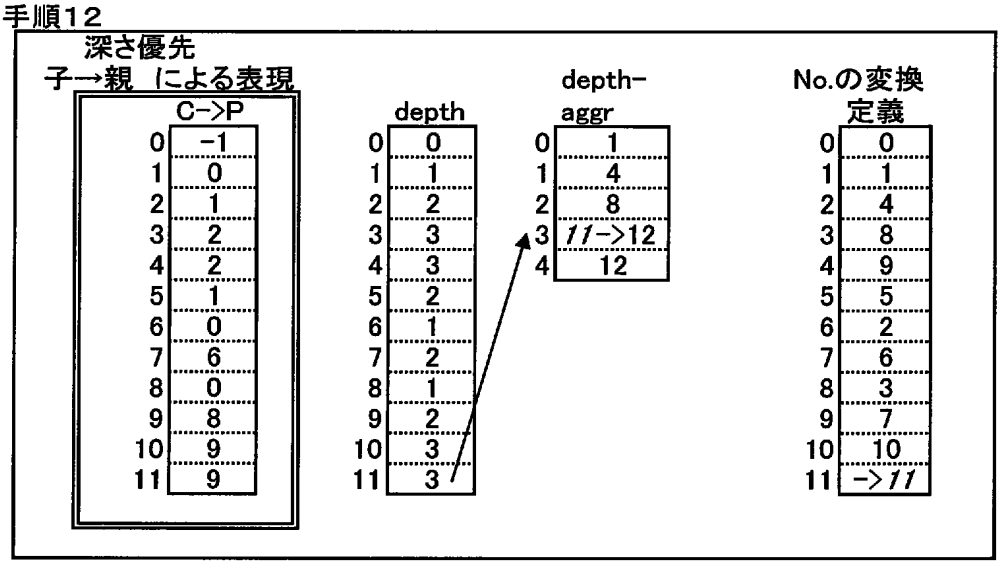


Fig.27C



[図28]

Fig.28


手順12
深さ優先
子→親 による表現
C->P
0 -1
1 0
2 1
3 2
4 2
5 1
6 0
7 6
8 0
9 8
10 9
11 9
depth
0 0
1 1
2 2
3 3
4 3
5 2
6 1
7 2
8 1
9 2
10 3
11 3
depth-
aggr
0 1
1 4
2 8
3 11->12
4 12
No.の変換
定義
0 0
1 1
2 4
3 8
4 9
5 5
6 2
7 6
8 3
9 7
10 10
11 ->11

[図29]

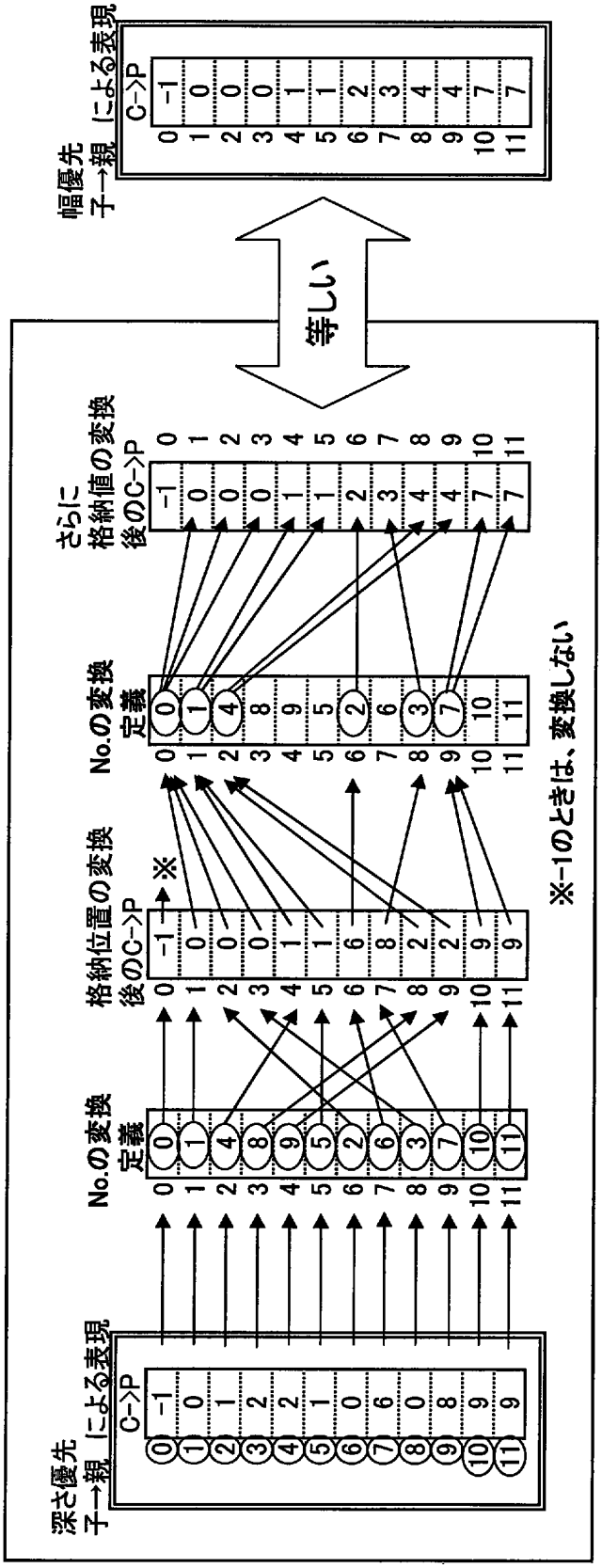

深さ優先
子→親 による表現
C->P
No.の変換
定義
格納位置の変換
後のC->P
※
No.の変換
定義
さらに
格納値の変換
後のC->P
等しい
幅優先
子→親 による表現
C->P
※-1のときは、変換しない

Fig.29

[図30]

Fig.30A

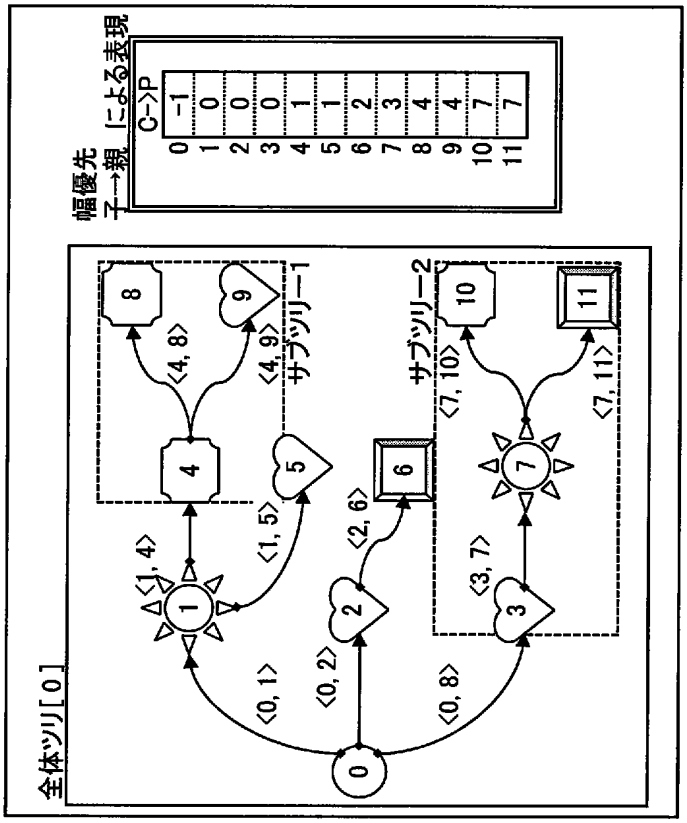
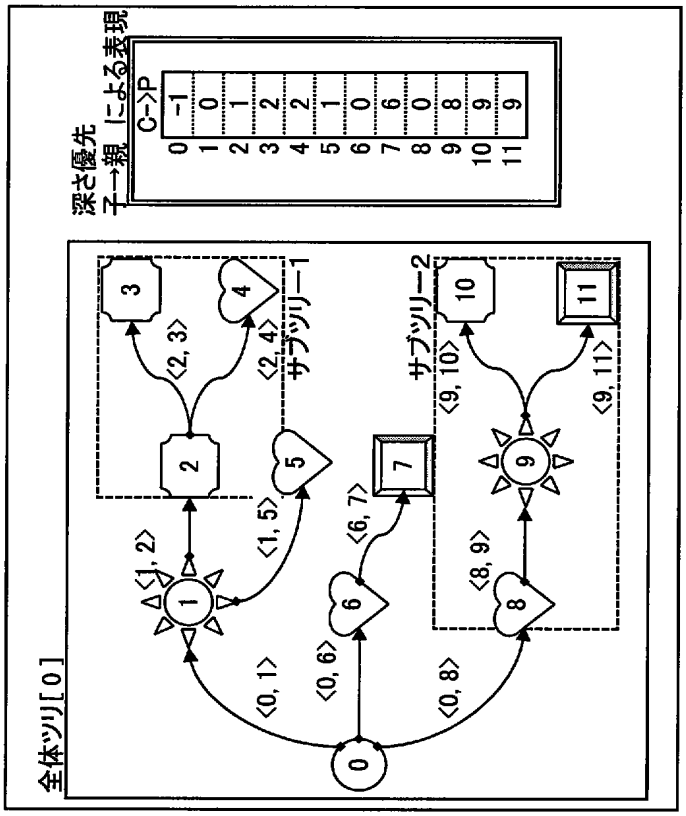
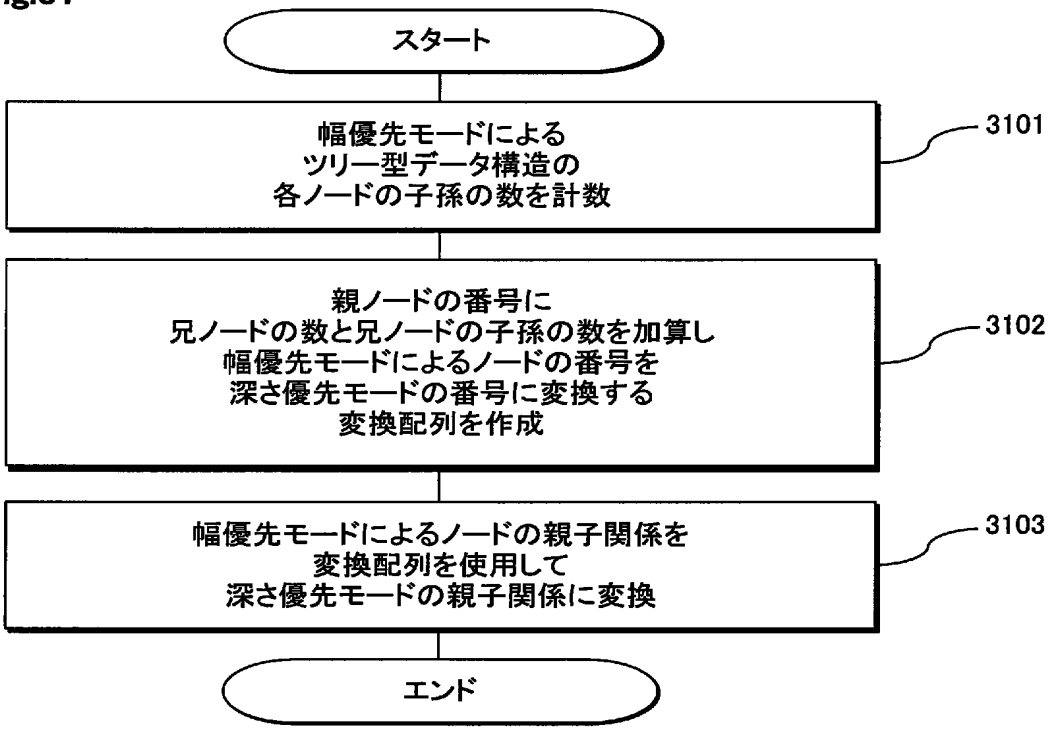


Fig.30B



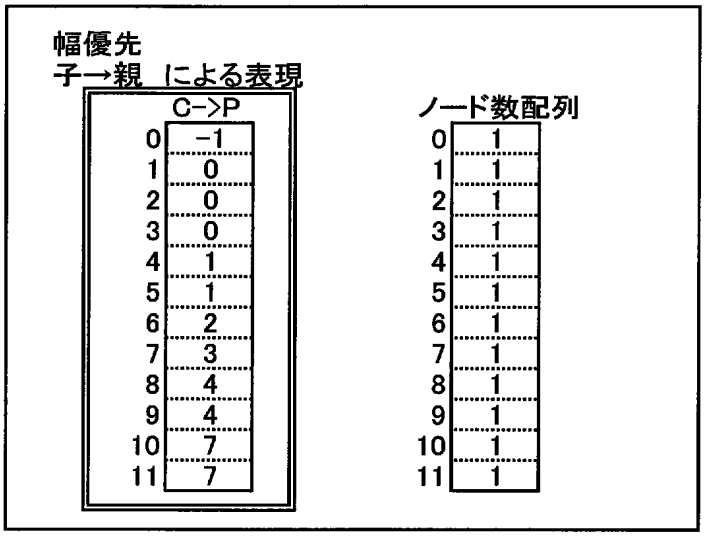
[図31]

Fig.31



[図32]

Fig.32



[図33]

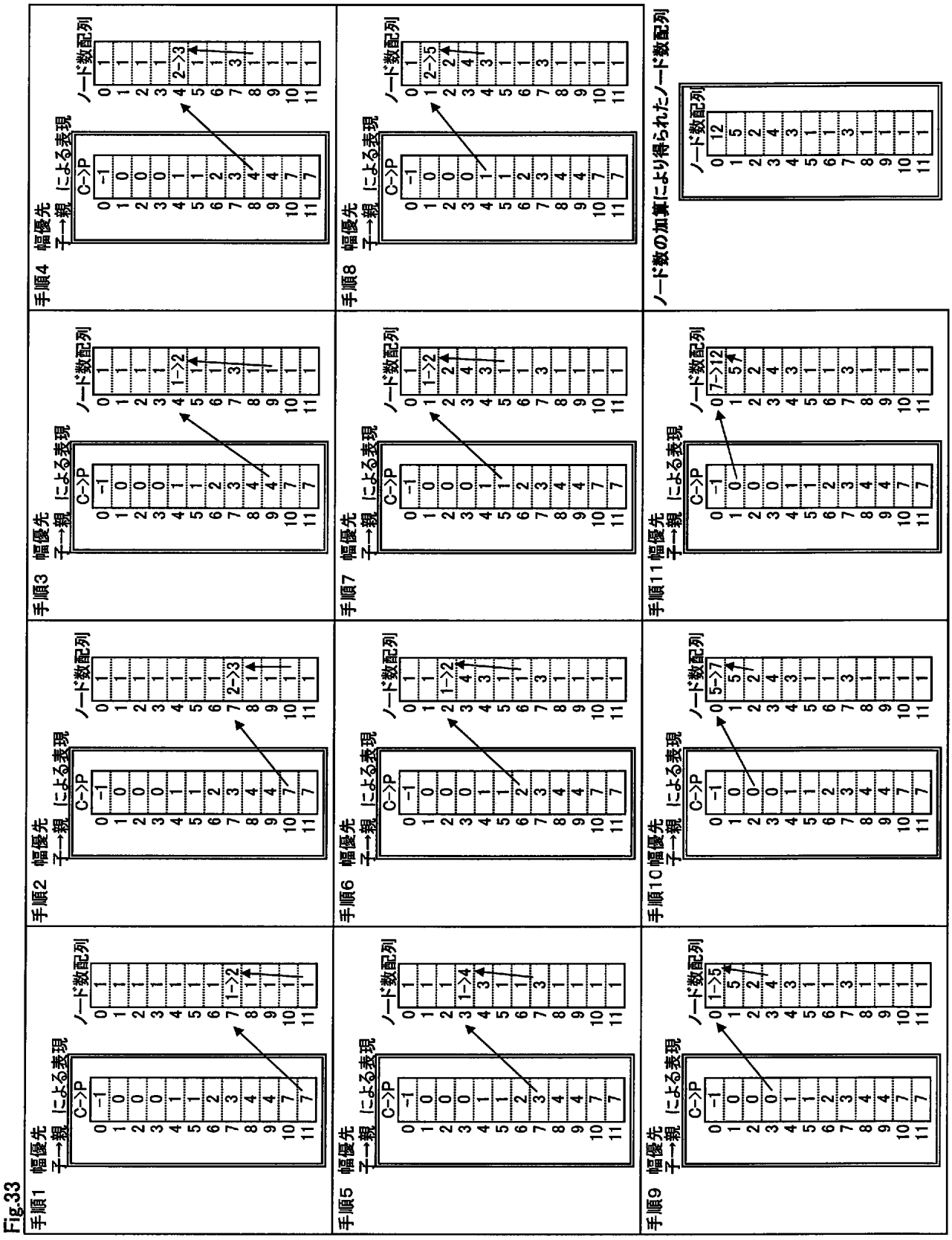
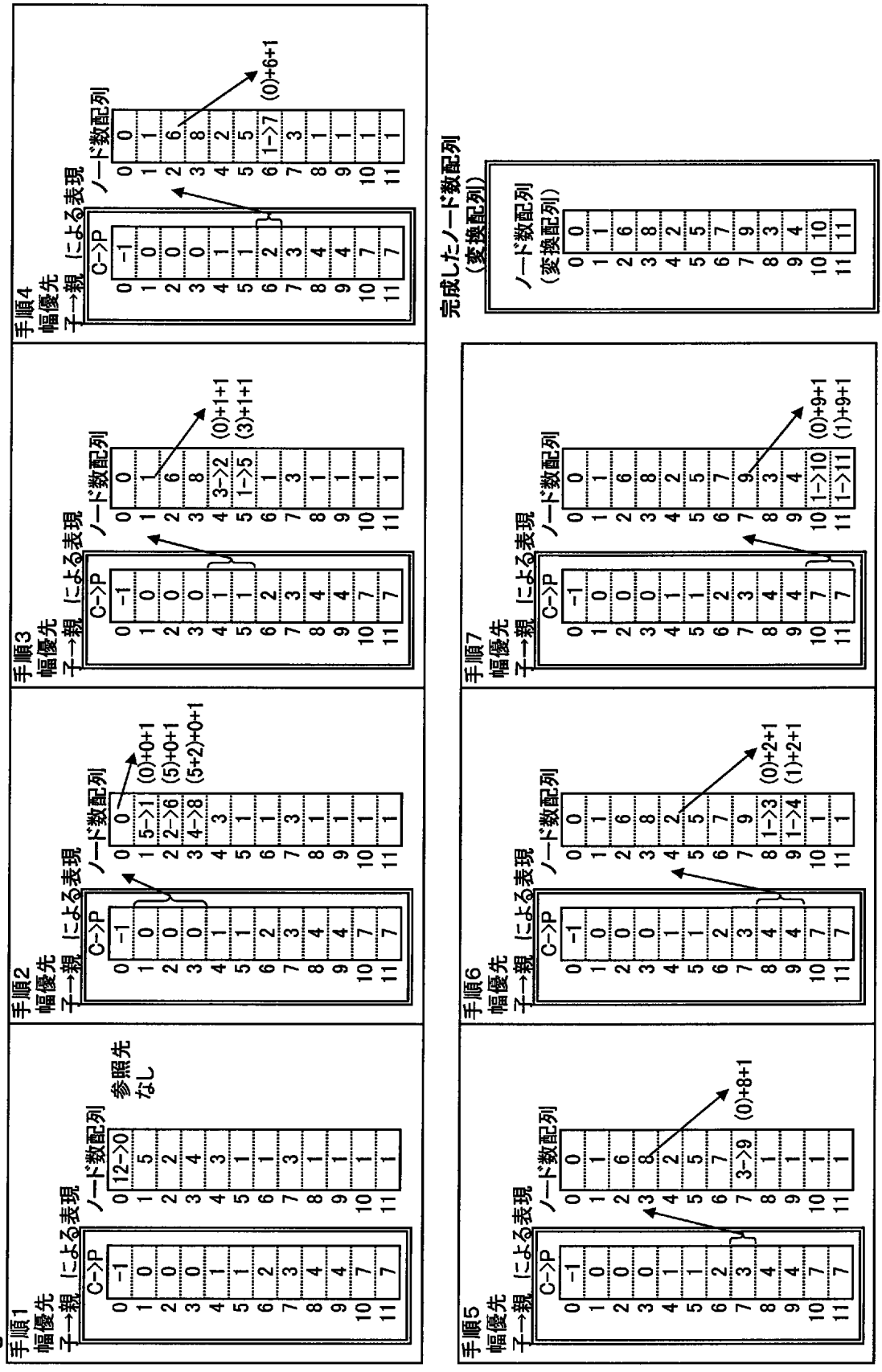


Fig.33
手順1 幅優先 子→親 による表現
手順2 幅優先 子→親 による表現
手順3 幅優先 子→親 による表現
手順4 幅優先 子→親 による表現
手順5 幅優先 子→親 による表現
手順6 幅優先 子→親 による表現
手順7 幅優先 子→親 による表現
手順8 幅優先 子→親 による表現
手順9 幅優先 子→親 による表現
手順10 幅優先 子→親 による表現
手順11 幅優先 子→親 による表現
C->P
ノード数配列
1->2
1->4
1->5
2->3
2->5
5->7
7->12
ノード数の加算により得られたノード数配列

[図34]

Fig.34


手順1
幅優先
子→親 による表現
C->P
ノード数配列
12->0
参照先
なし
手順2
幅優先
子→親 による表現
C->P
ノード数配列
5->1
2->6
4->8
(0)+0+1
(5)+0+1
(5+2)+0+1
手順3
幅優先
子→親 による表現
C->P
ノード数配列
3->2
1->5
(0)+1+1
(3)+1+1
手順4
幅優先
子→親 による表現
C->P
ノード数配列
1->7
(0)+6+1
手順5
幅優先
子→親 による表現
C->P
ノード数配列
3->9
(0)+8+1
手順6
幅優先
子→親 による表現
C->P
ノード数配列
1->3
1->4
(0)+2+1
(1)+2+1
手順7
幅優先
子→親 による表現
C->P
ノード数配列
1->10
1->11
(0)+9+1
(1)+9+1
完成したノード数配列
(変換配列)
ノード数配列
(変換配列)

[図35]

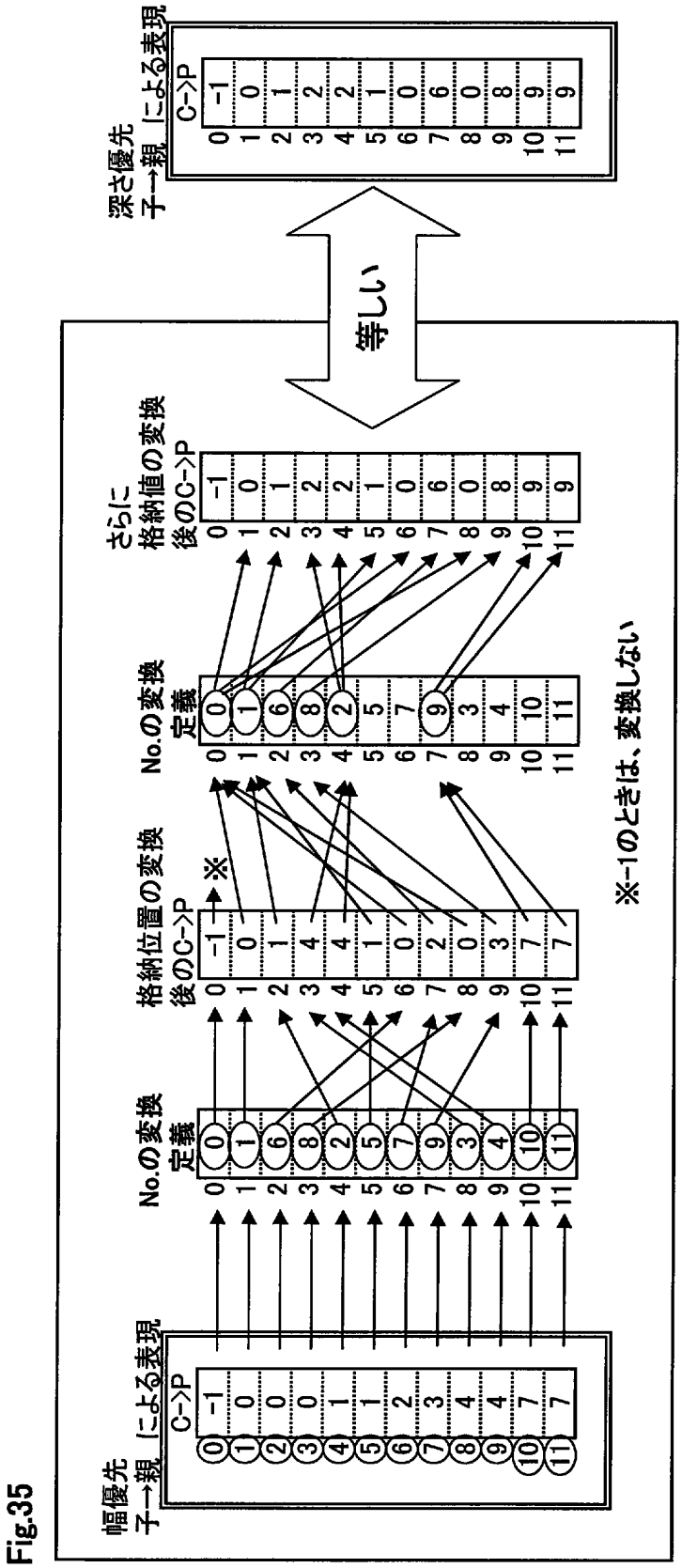
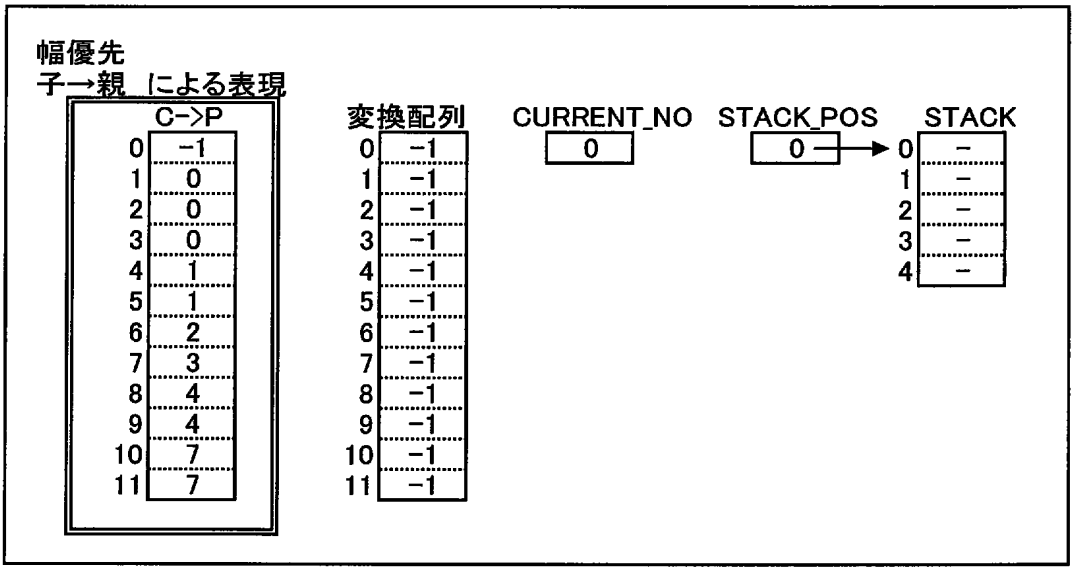

幅優先
子→親 による表現
C->P
No.の変換
定義
格納位置の変換
後のC->P
※
※-1のときは、変換しない
さらに
格納値の変換
後のC->P
No.の変換
定義
等しい
深さ優先
子→親 による表現
C->P

Fig.35

[図36]

Fig.36


幅優先
子→親 による表現
C->P
0 -1
1 0
2 0
3 0
4 1
5 1
6 2
7 3
8 4
9 4
10 7
11 7
変換配列
0 -1
1 -1
2 -1
3 -1
4 -1
5 -1
6 -1
7 -1
8 -1
9 -1
10 -1
11 -1
CURRENT_NO
0
STACK_POS
0
STACK
0 -
1 -
2 -
3 -
4 -

[図37]

Fig.37A

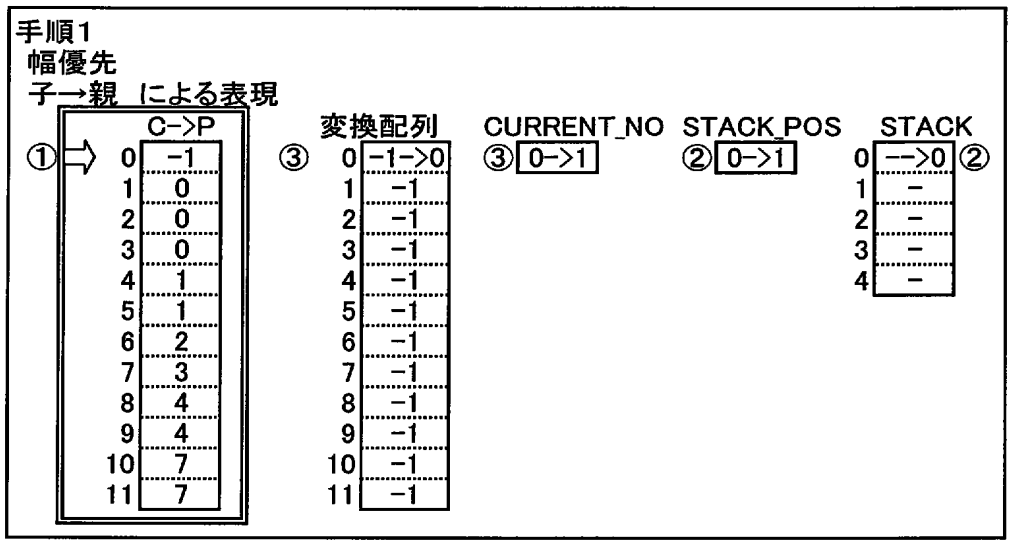


Fig.37B

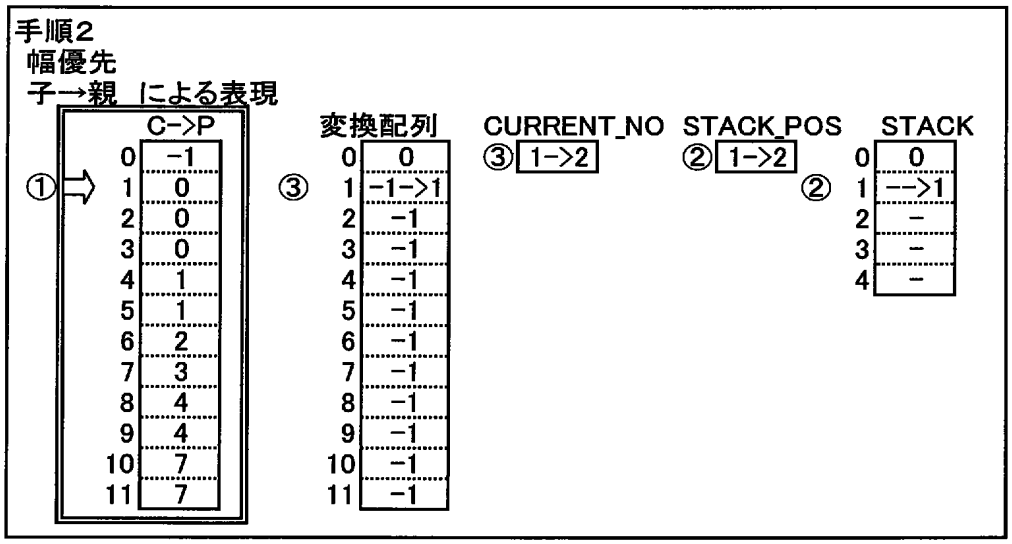
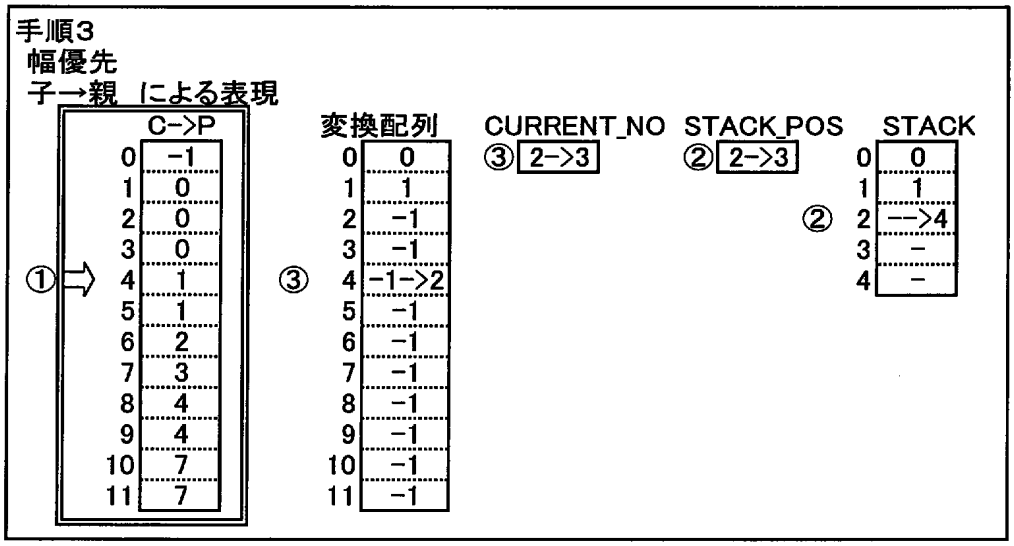


Fig.37C



[図38]

Fig.38A

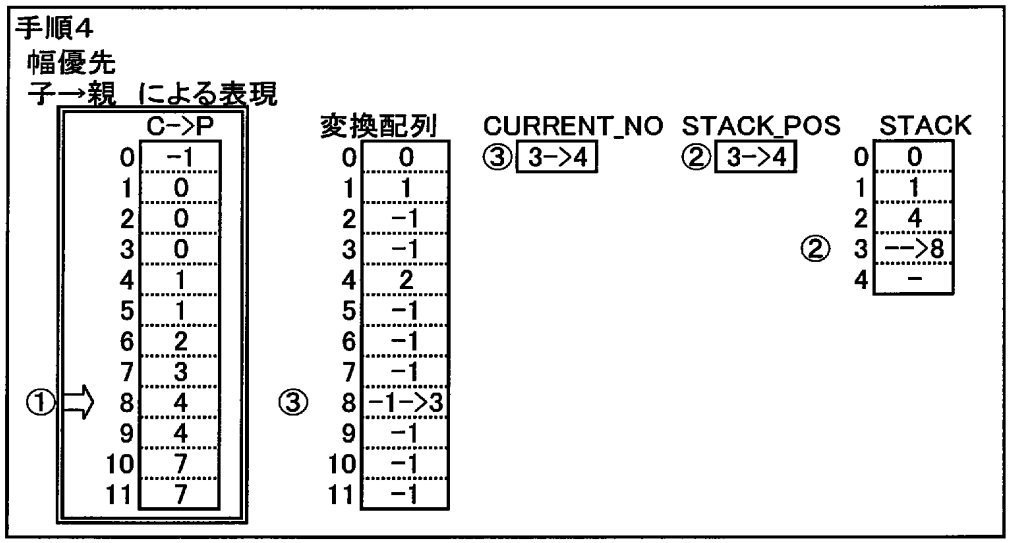


Fig.38B

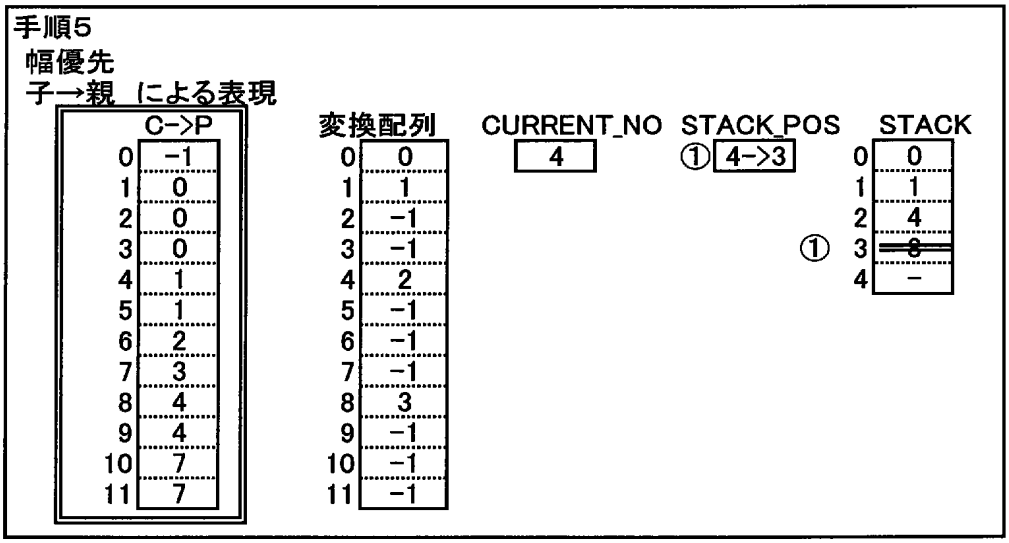
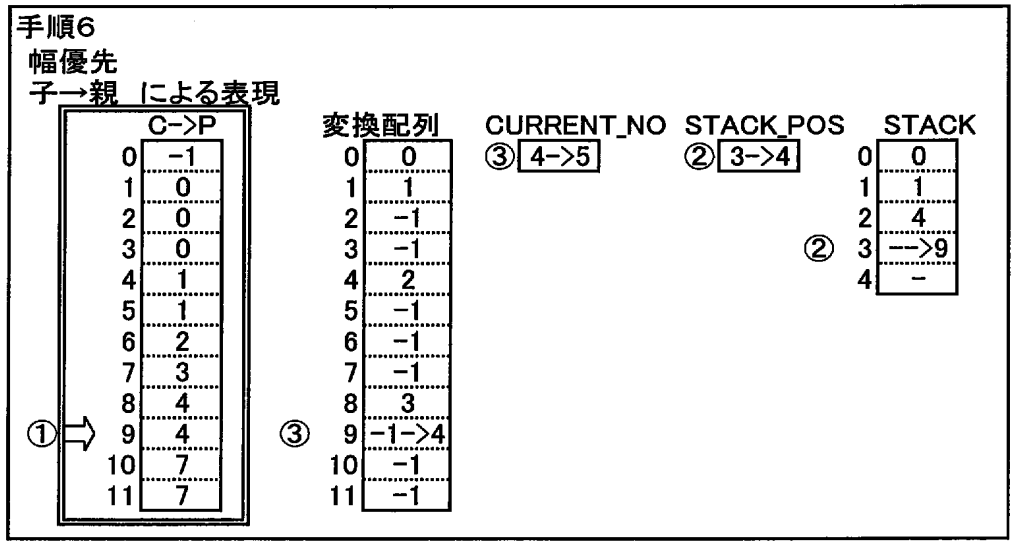


Fig.38C



[図39]

Fig.39A

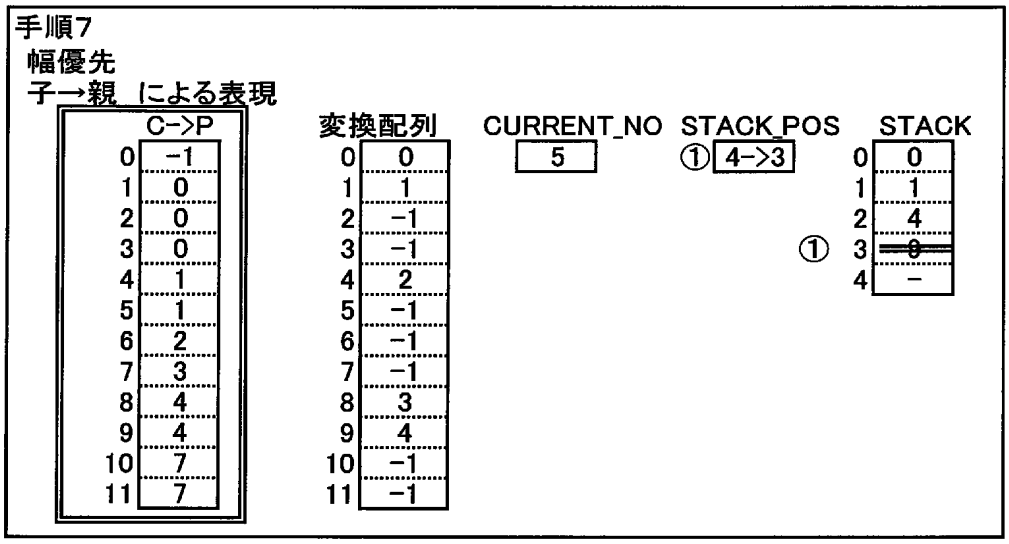


Fig.39B

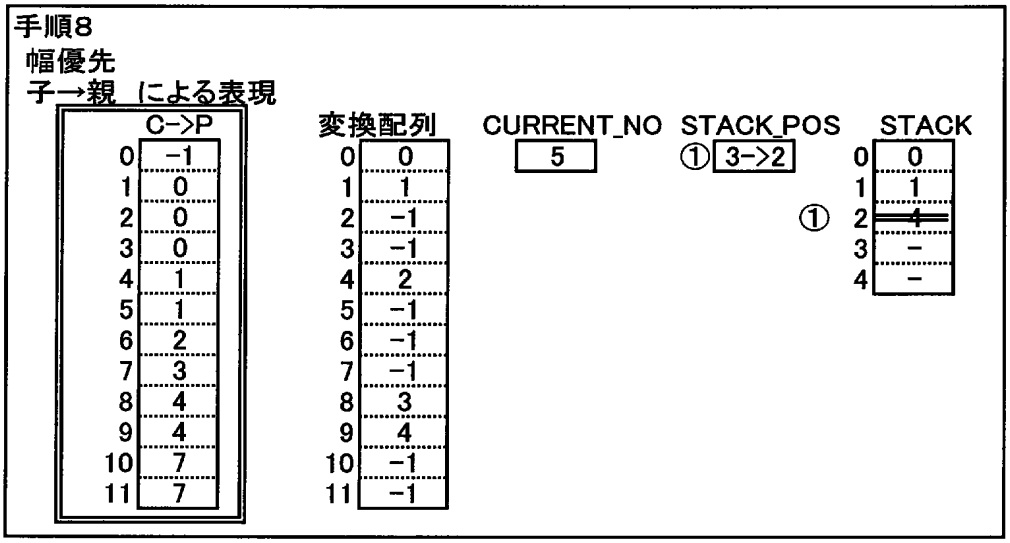
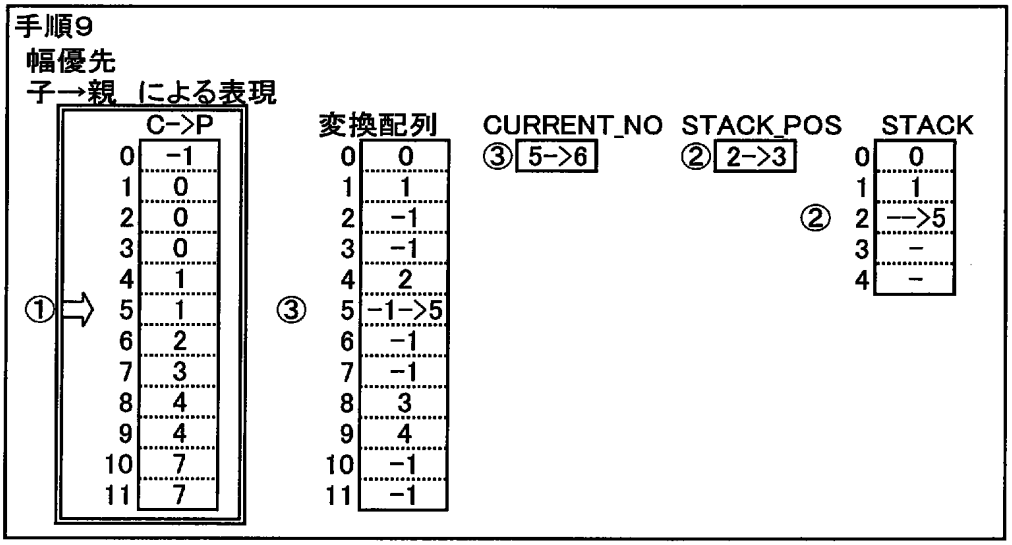


Fig.39C



[図40]

Fig.40A

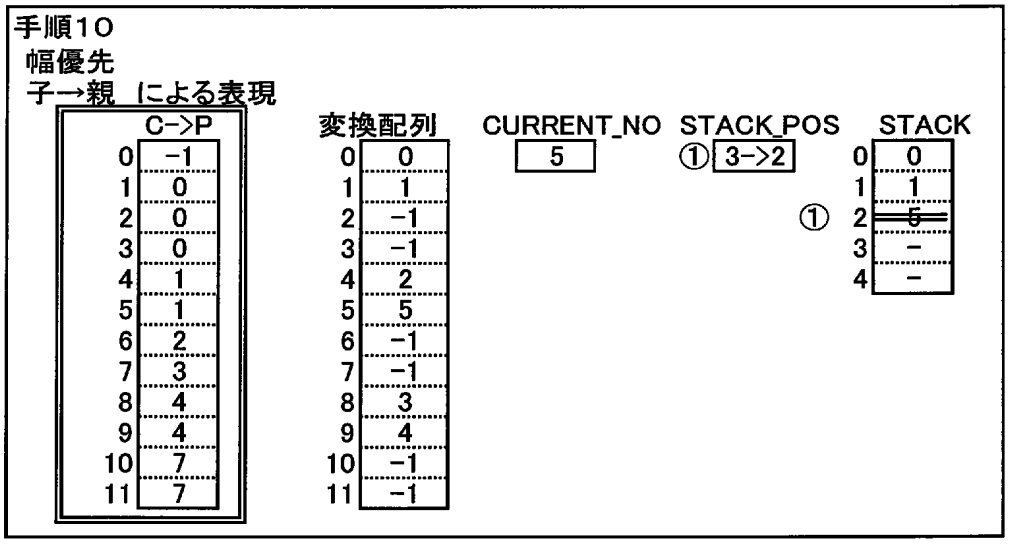


Fig.40B

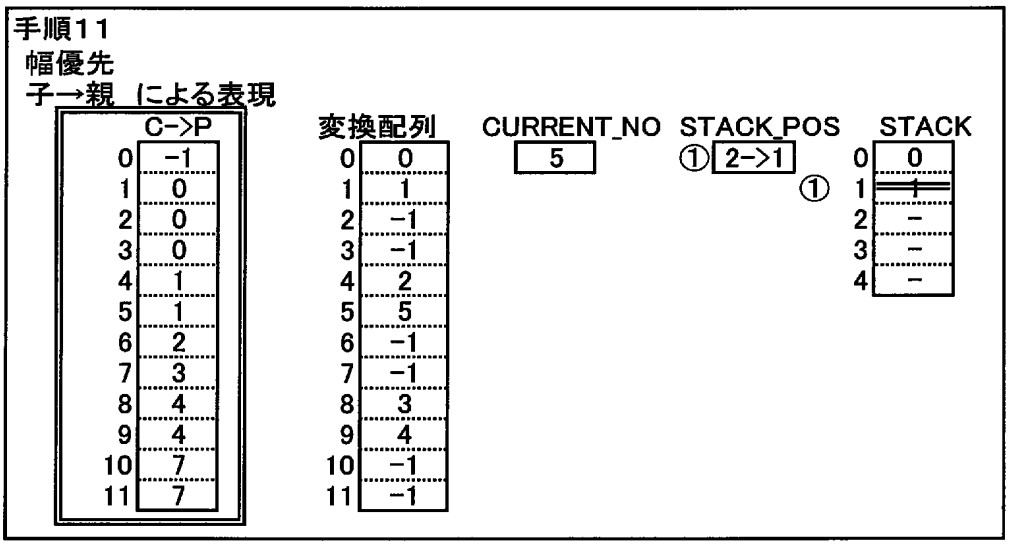
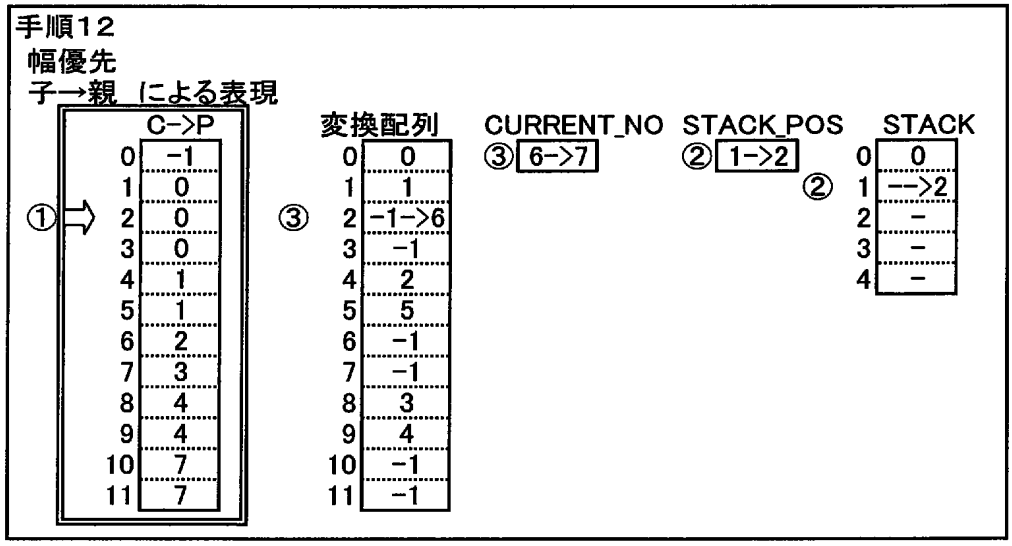


Fig.40C



[図41]

Fig.41A

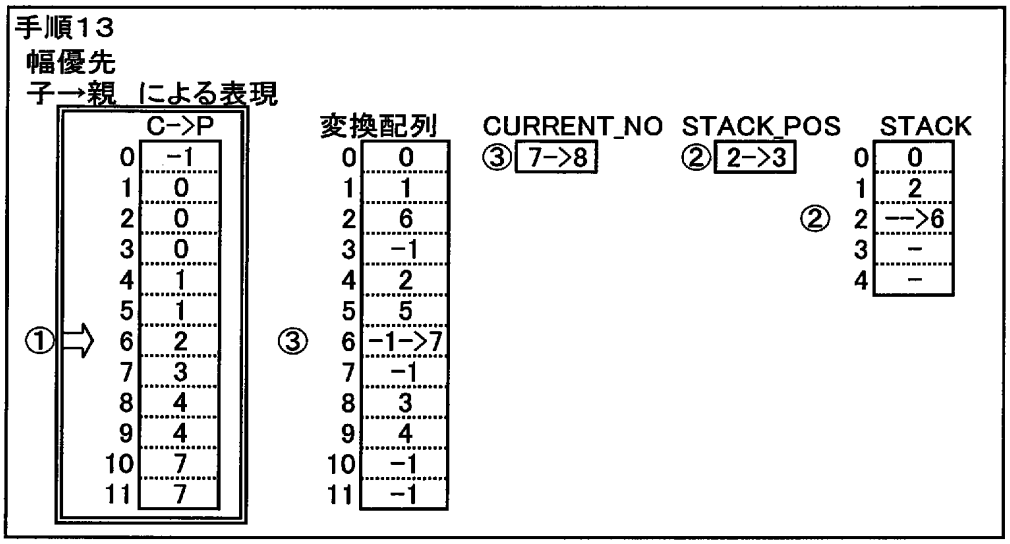


Fig.41B

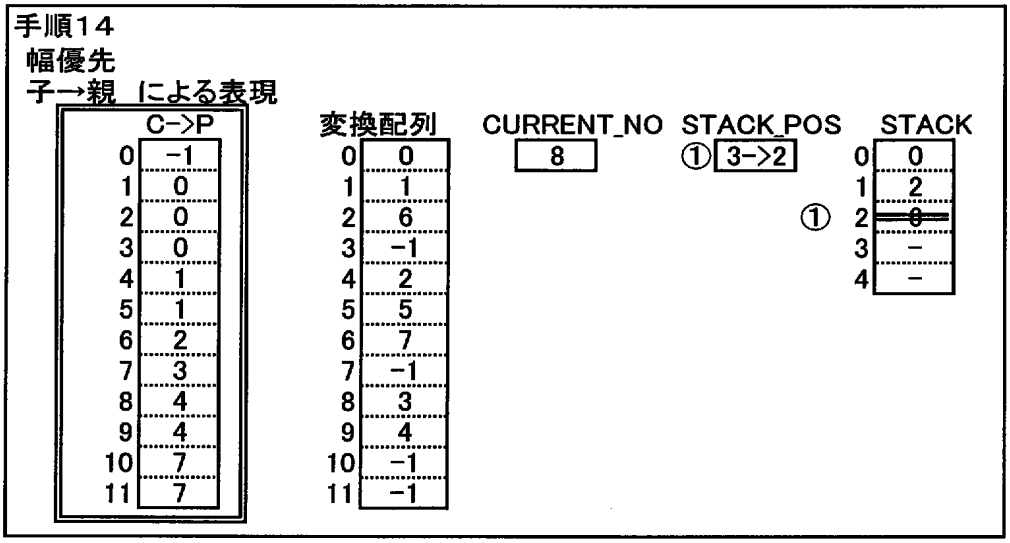
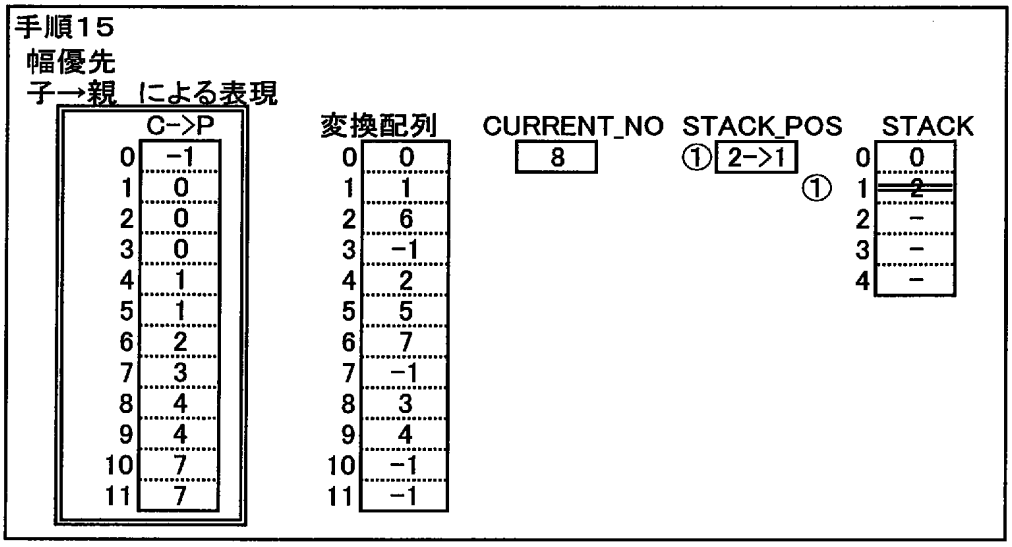


Fig.41C



[図42]

Fig.42A

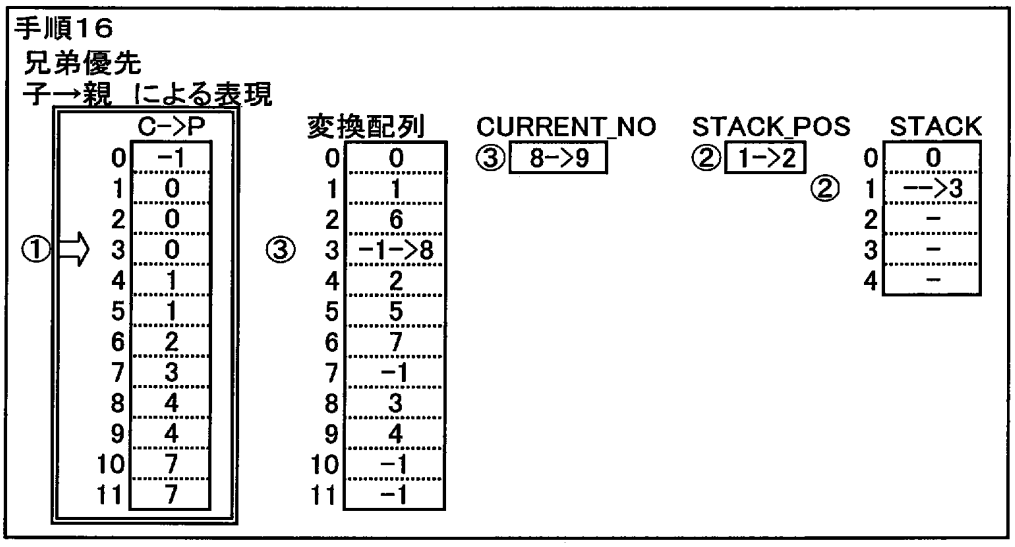


Fig.42B

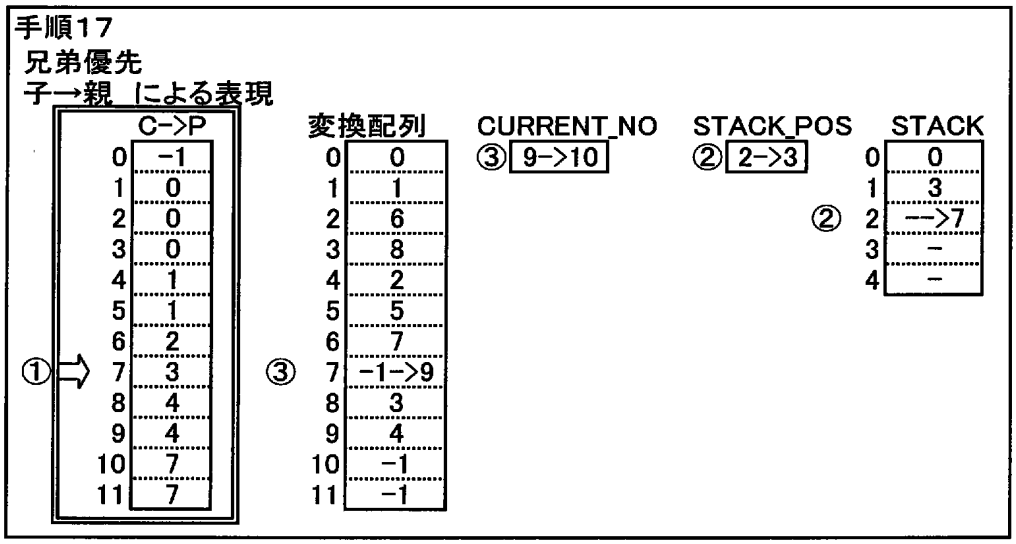
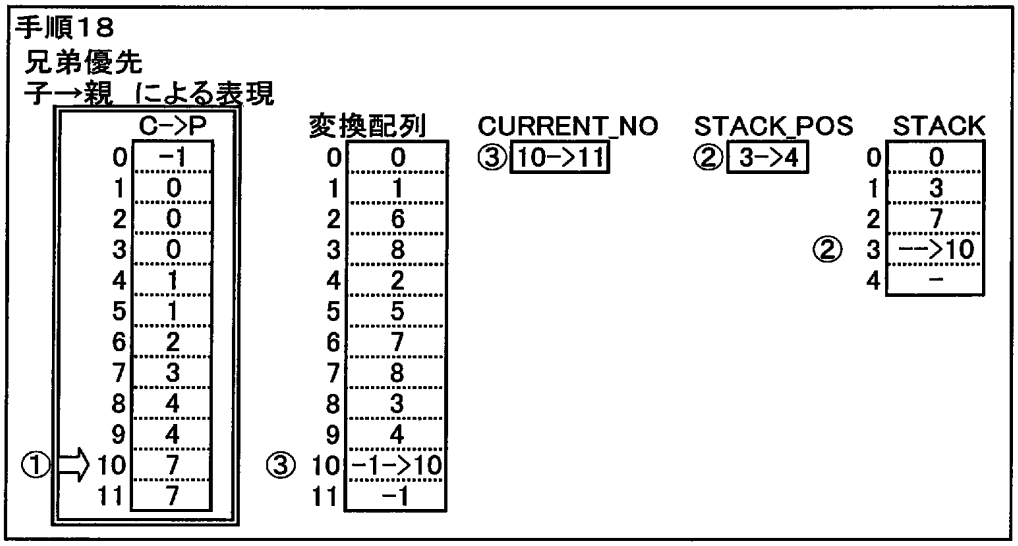


Fig.42C



[図43]

Fig.43A

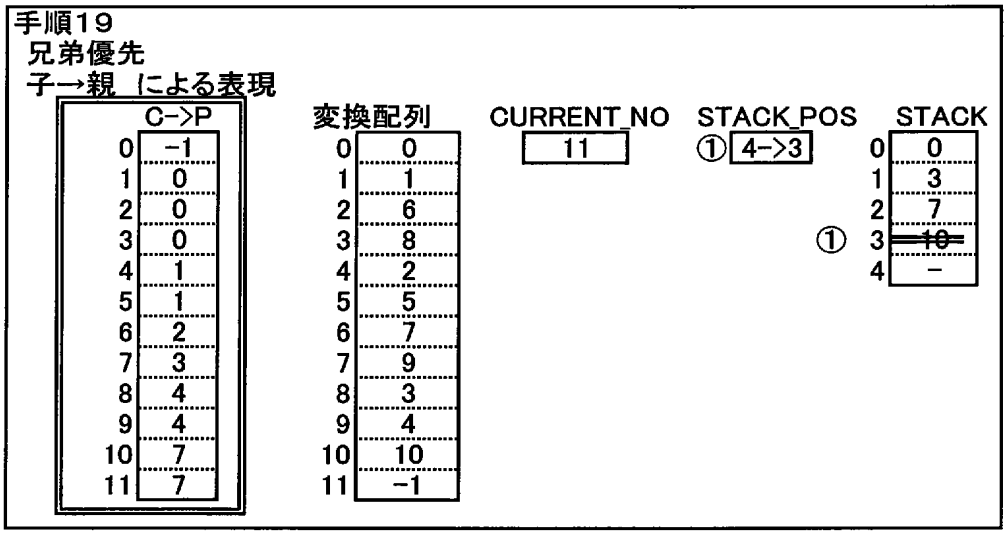
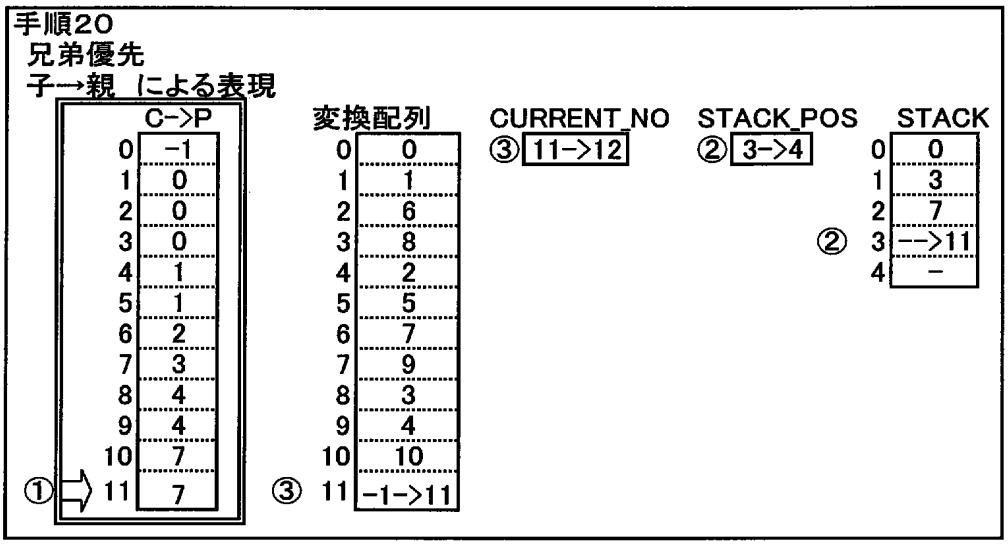
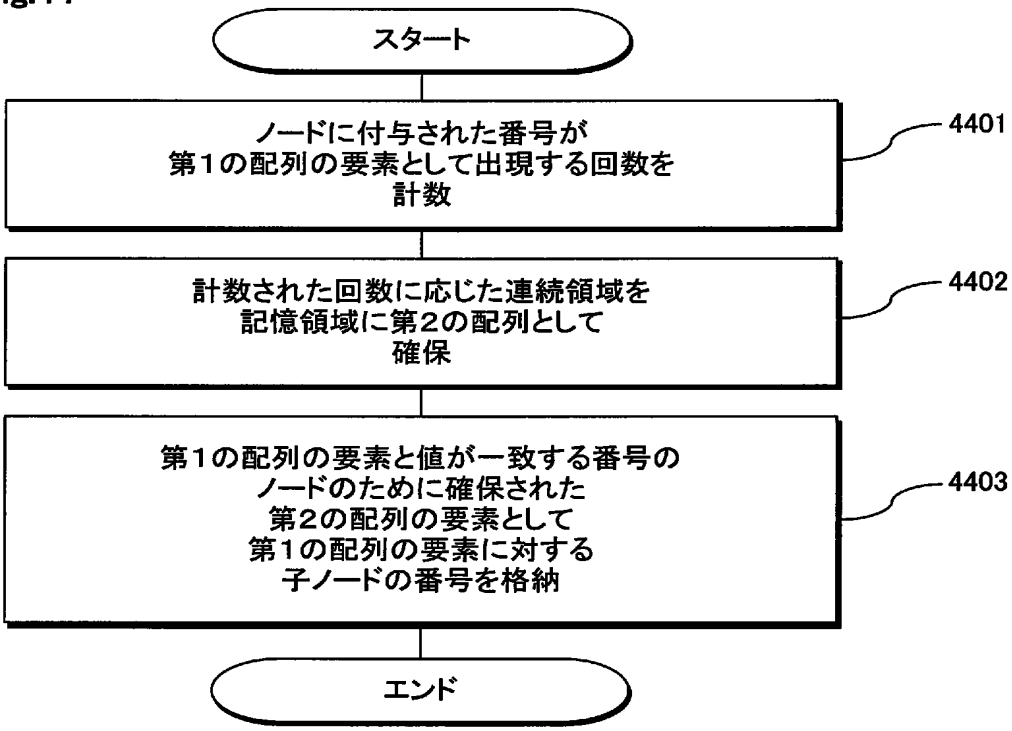


Fig.43B



[図44]

Fig.44


スタート
ノードに付与された番号が
第１の配列の要素として出現する回数を
計数
4401
計数された回数に応じた連続領域を
記憶領域に第２の配列として
確保
4402
第１の配列の要素と値が一致する番号の
ノードのために確保された
第２の配列の要素として
第１の配列の要素に対する
子ノードの番号を格納
4403
エンド

[図45]

Fig.45A

全体ツリー

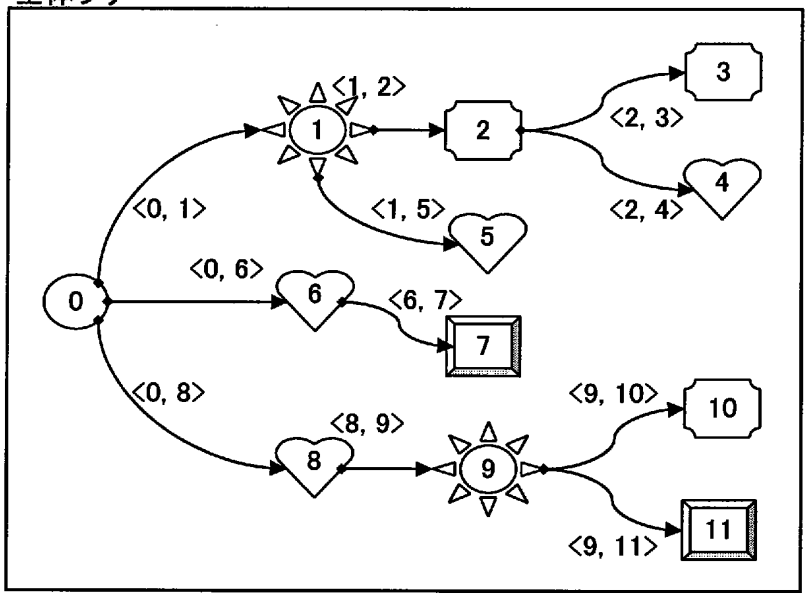


Fig.45B

子→親 による表現

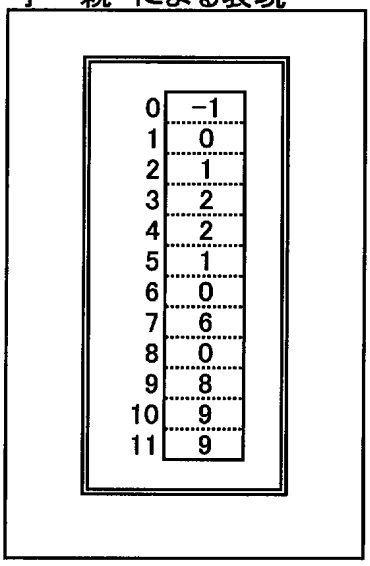
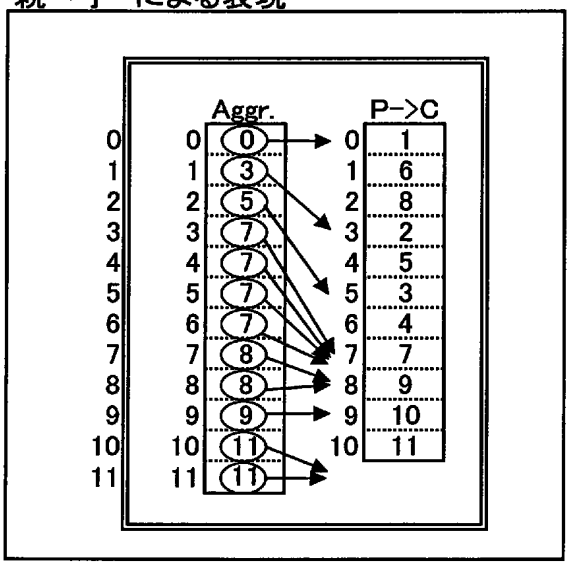


Fig.45C

親→子 による表現



[図46]

Fig.46A

手順1：領域確保と初期化

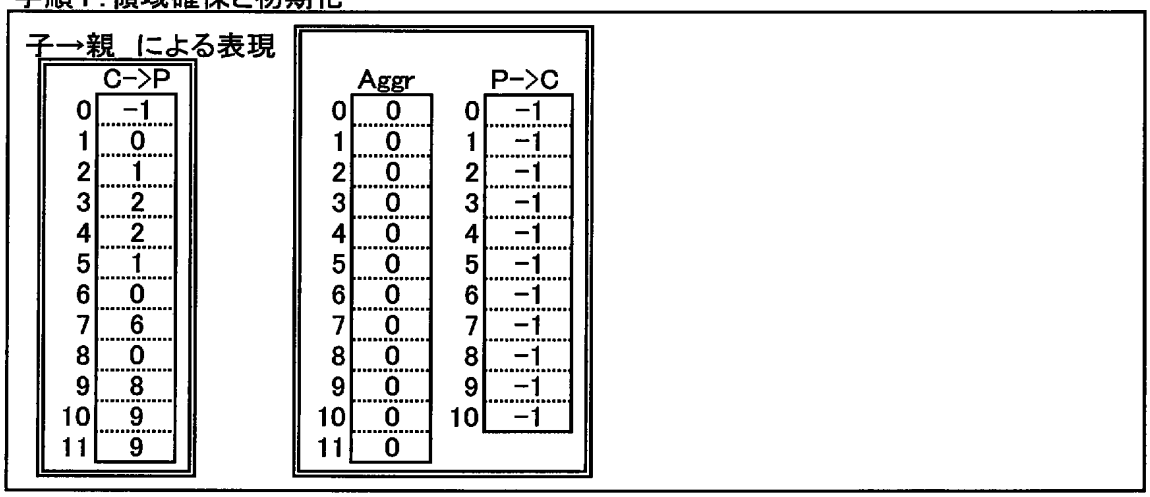


Fig.46B

手順2 存在数をカウント

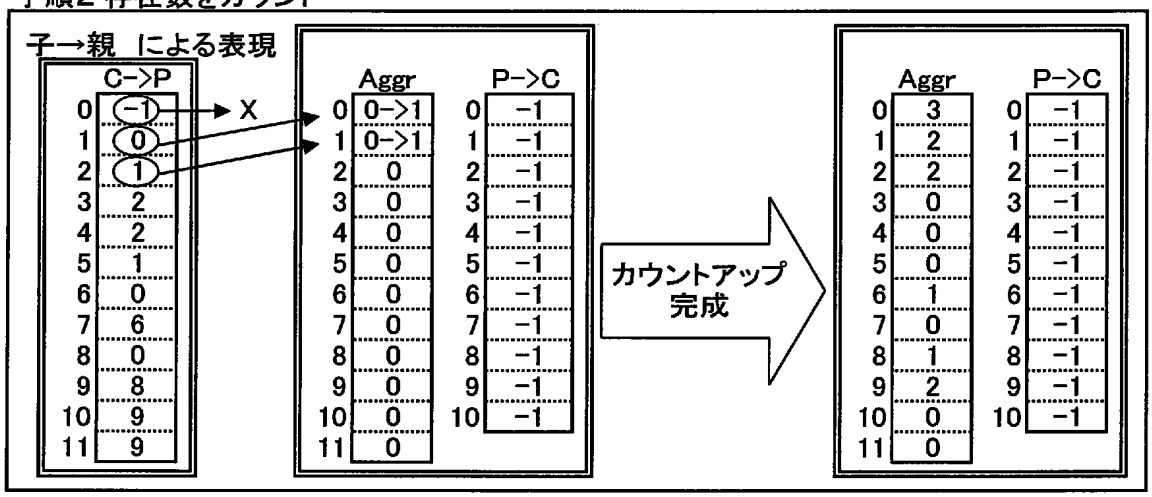
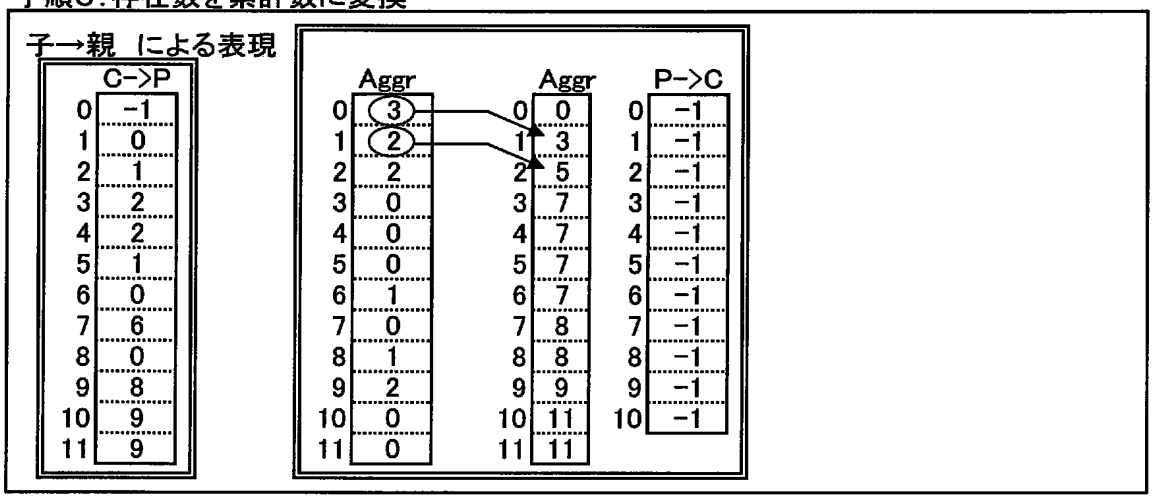


Fig.46C

手順3：存在数を累計数に変換



[図47]

Fig.47A

手順4:ノード番号を転送する

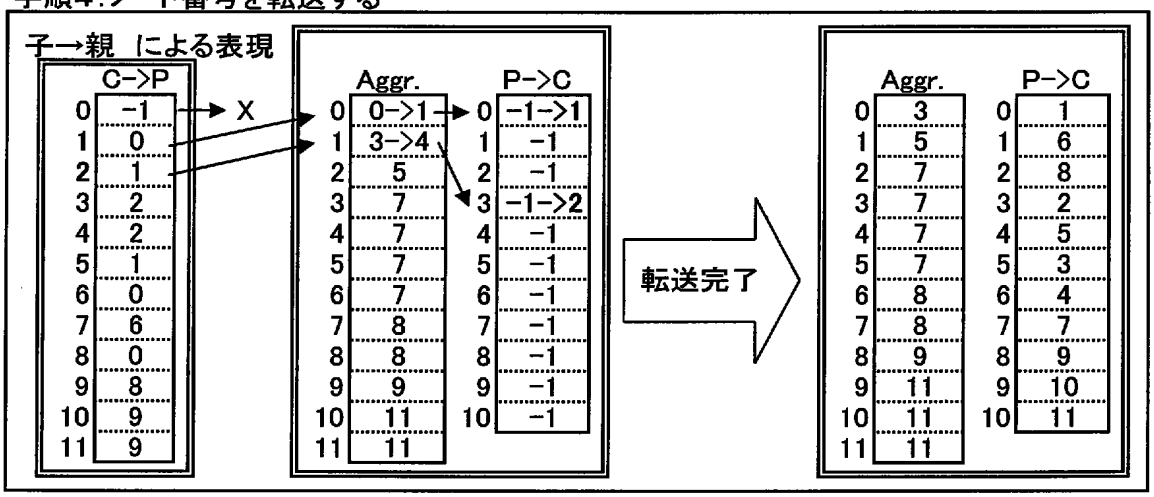


Fig.47B

手順5:配列Aggrをノード番号転送前の状態に戻す

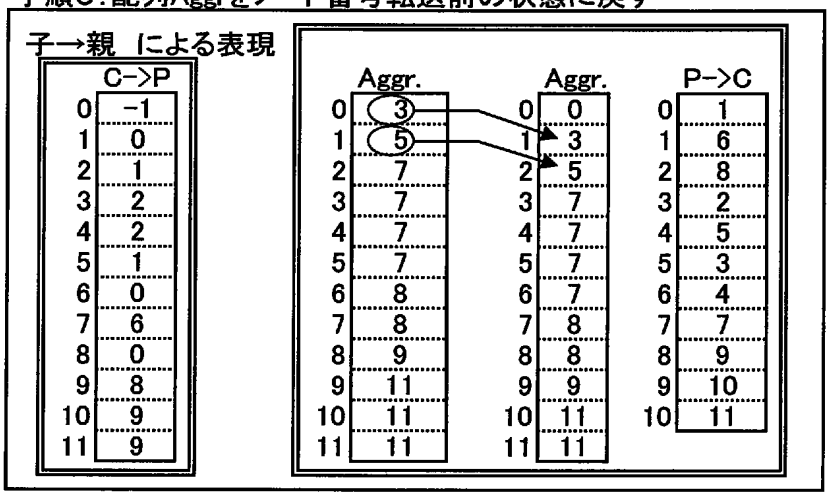
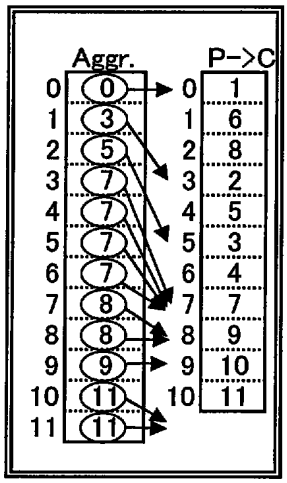


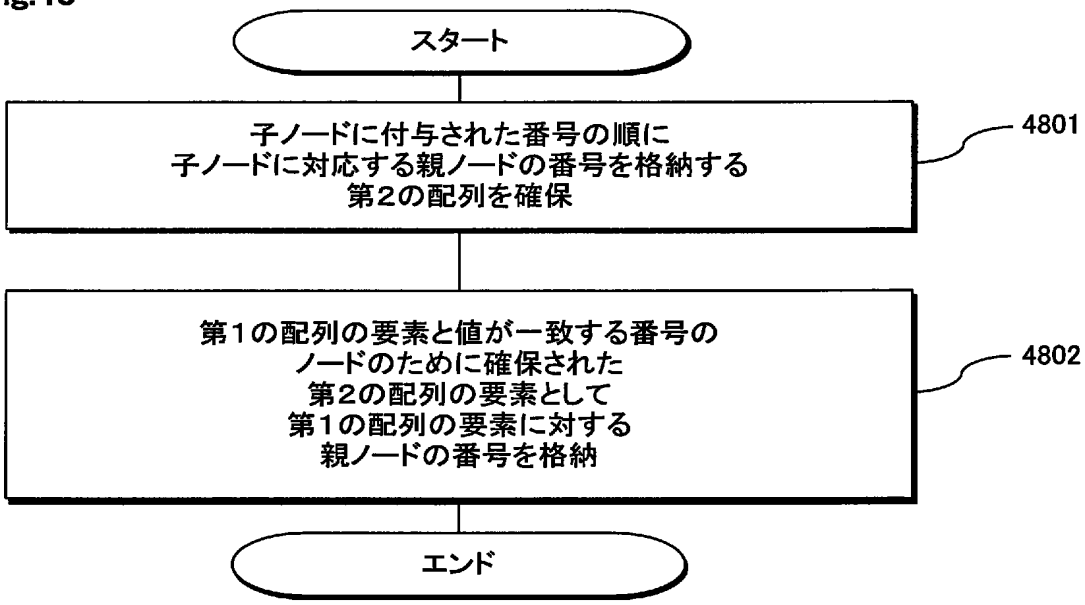
Fig.47C

変換結果



[図48]

Fig.48


スタート
子ノードに付与された番号の順に
子ノードに対応する親ノードの番号を格納する
第2の配列を確保
4801
第1の配列の要素と値が一致する番号の
ノードのために確保された
第2の配列の要素として
第1の配列の要素に対する
親ノードの番号を格納
4802
エンド

[図49]

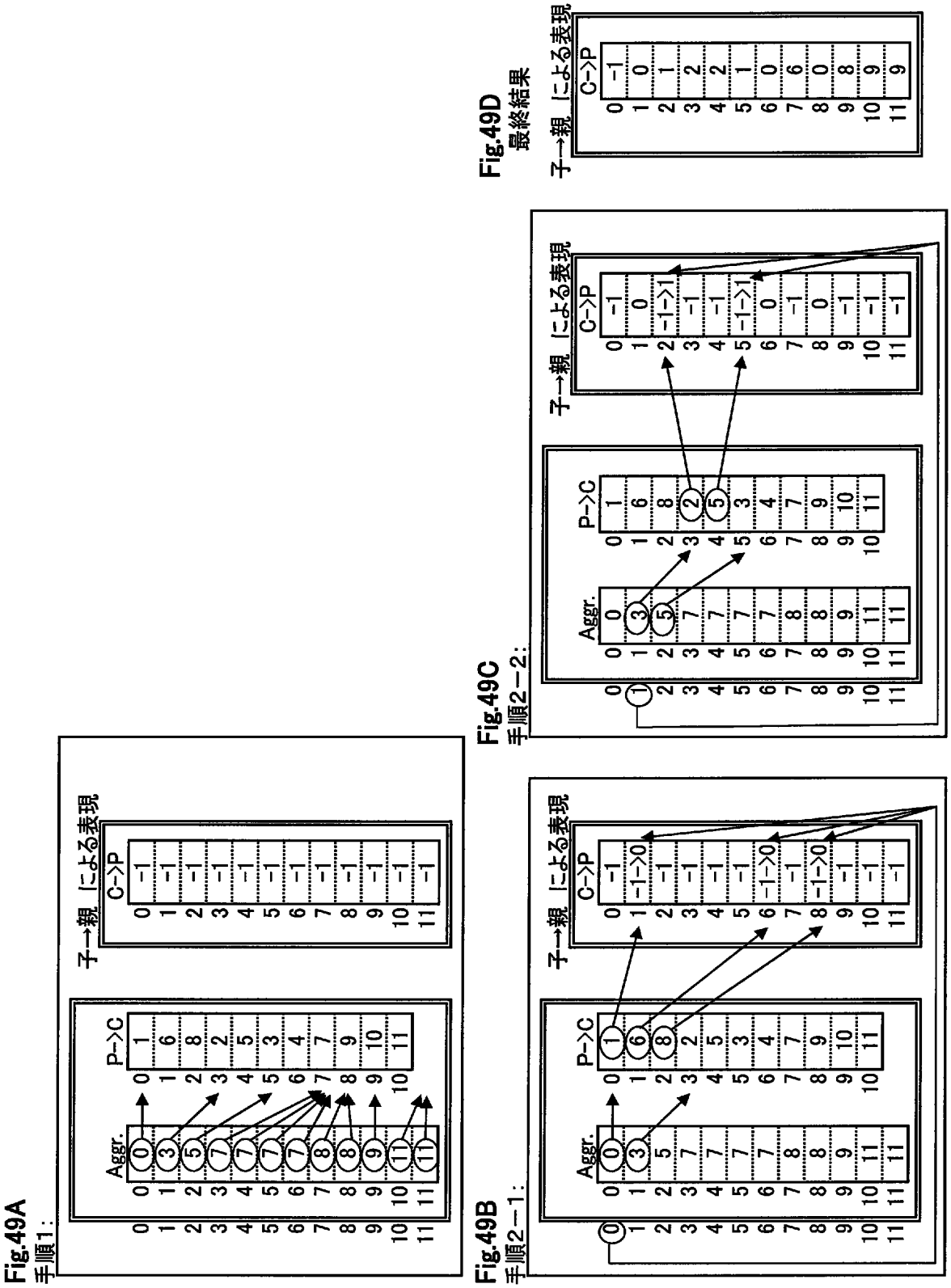
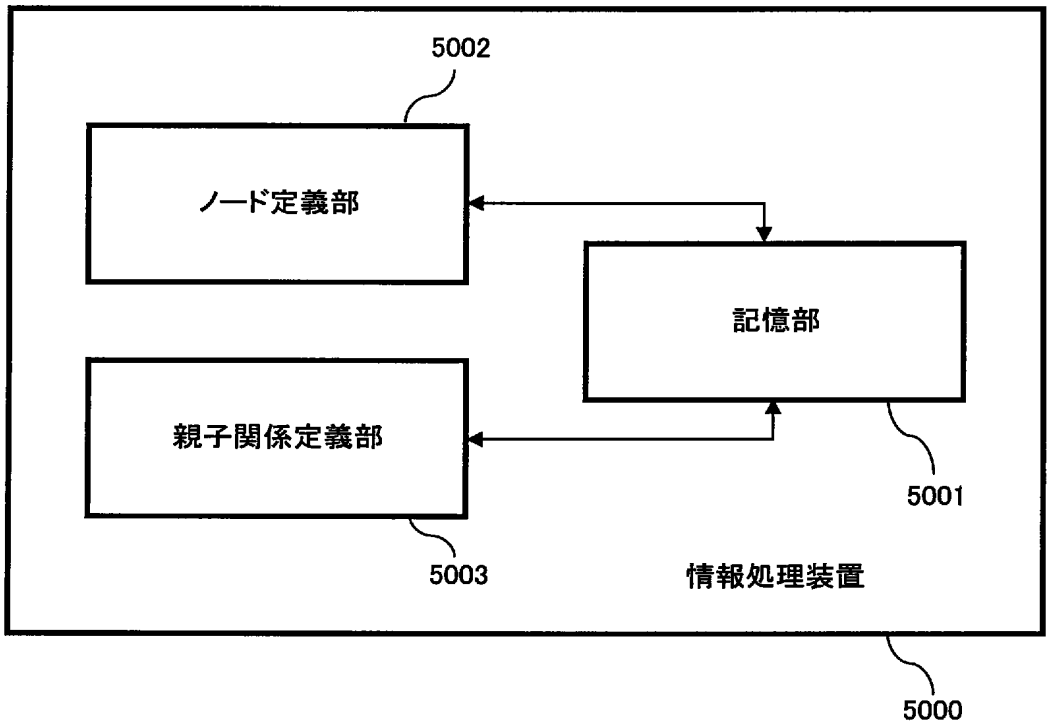


Fig.49A
手順1:
子→親 による表現
C->P
P->C
Aggr.
Fig.49B
手順2−1:
Fig.49C
手順2−2:
Fig.49D
最終結果

[図50]

Fig.50



INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2005/004190

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl[7] G06F17/30

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl[7] G06F17/30

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1922-1996 Jitsuyo Shinan Toroku Koho 1996-2005
Kokai Jitsuyo Shinan Koho 1971-2005 Toroku Jitsuyo Shinan Koho 1994-2005

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
JICST FILE(JOIS)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	R. Sedgwick, translated by Kohei NOSHITA et al., Algorithm Vol.1, =Kiso· Seiretsu, first edition, Kindai Kagaku Sha, 1990, pages 39 to 55	1-54
X	JP 10-240741 A (Nippon Telegraph And Telephone Corp.), 11 September, 1998 (11.09.98), Par. Nos. [0001] to [0006]; Figs. 3 to 5 (Family: none)	1-54
A	JP 11-327993 A (Xerox Corp.), 30 November, 1999 (30.11.99), Full text; Figs. 1 to 30 & US 2002/0067360 A1 & EP 0950960 A2	1-54

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
08 April, 2005 (08.04.05)	26 April, 2005 (26.04.05)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2005/004190

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 2000-99226 A (Xerox Corp.), 07 April, 2000 (07.04.00), Full text; Figs. 1 to 10 & US 2001/0045952 A1 & EP 0977154 A2	1-54

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００５／００４１９０

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.[7]　G06F17/30

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.[7]　G06F17/30

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２００５年
日本国実用新案登録公報	１９９６－２００５年
日本国登録実用新案公報	１９９４－２００５年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

JICST ファイル(JOIS)

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
X	Ｒ．セジウィック著，野下浩平、外３名訳，アルゴリズム　第１巻＝基礎・整列，初版，近代科学社，１９９０，ｐ．３９－５５	１－５４
X	JP １０－２４０７４１ Ａ（日本電信電話株式会社）１９９８．０９．１１，第１－６段落、第３－５図（ファミリーなし）	１－５４
A	JP １１－３２７９９３ Ａ（ゼロックス　コーポレイション）１９９９．１１．３０，全文，第１－３０図 ＆ ＵＳ ２００２／００６７３６０ Ａ１ ＆ ＥＰ ０９５０９６０ Ａ２	１－５４

☑ Ｃ欄の続きにも文献が列挙されている。　　☐ パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日 ０８．０４．２００５	国際調査報告の発送日 26.4.2005
国際調査機関の名称及びあて先 日本国特許庁（ＩＳＡ／ＪＰ） 郵便番号１００－８９１５ 東京都千代田区霞が関三丁目４番３号	特許庁審査官（権限のある職員）　5M　３２５２ 水野　恵雄 電話番号　０３－３５８１－１１０１　内線　３５９７

Ｃ（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
A	JP ２０００－９９２２６ A (ゼロックス　コーポレイション) ２０００.０４.０７, 全文, 第１－１０図 ＆ ＵＳ ２００１／００４５９５２ Ａ１ ＆ ＥＰ ０９７７１５４ Ａ２	１－５４